

CONTROL DATA
CORPORATION

CDC® CYBER 180 MODELS 810 AND 830

AA161-A CENTRAL COMPUTER

PLUS OPTIONS

=====

THEORY OF OPERATION

=====

HARDWARE MAINTENANCE MANUAL

REVISION RECORD

REVISION	DESCRIPTION
A (6-84)	Manual released. This edition obsoletes all previous editions.
B (5-85)	Manual revised; incorporates new memory and channel options.

Publication no.
60469460

© 1984, 1985
by Control Data Corporation

Address comments concerning
this manual to:

Control Data Canada
Toronto Publications
1855 Minnesota Court
MISSISSAUGA, Ont.,
Canada L5N 1K7

or use Comment Sheet in
the back of this manual

MANUAL TO EQUIPMENT LEVEL CORRELATION SHEET

=====

This manual reflects the equipment configuration listed below.

EXPLANATION: Locate the equipment type and series number, as shown on the equipment FCO log, in the list below. Immediately to the right of the series number is an FCO number. If that number and all of the numbers underneath it match all of the numbers on the equipment FCO log, then this manual accurately reflects the equipment.

EQUIPMENT TYPE	SERIES	WITH FCOs	COMMENTS
AA161-A	79		
AT453-A	10		
AT462-A	06		
AT463-A	09		
AU125-A	04		
AU127-A	36		
AU127-B	37		
AU129-A	01		
AU130-A	01		
BS167-A	01		
BS222-A	01		

THIS PAGE INTENTIONALLY LEFT BLANK.

LIST OF EFFECTIVE PAGES

=====
New features, as well as changes, deletions, and additions to information in this manual, are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV
Cover	A	Tab	A	II-3-15	A	Divider	A
Title Page	A	Divider	A	Divider	A	II-6-1	B
2	B	II-1-1	B	II-4-1	A	II-6-2	A
3	B	II-1-2	B	II-4-2	B	II-6-3	B
4	A	II-1-3	B	II-4-3	B	II-6-4	B
5/6	B	II-1-4	B	II-4-4	A	II-6-5	A
7	B	II-1-5	B	II-4-5	A	II-6-6	A
8	A	II-1-6	B	II-4-6	A	II-6-7	A
9	B	Divider	A	II-4-7	A	II-6-8	A
10	A	II-2-1	A	II-4-8	B	II-6-9	A
11	A	II-2-2	A	II-4-9	A	II-6-10	A
12	A	II-2-3	A	II-4-10	A	II-6-11	A
13	A	II-2-4	A	II-4-11	A	II-6-12	A
14	A	II-2-5	A	II-4-12	A	II-6-13	A
15	A	II-2-6	A	II-4-13	A	II-6-14	A
16	B	II-2-7	A	II-4-14	A	II-6-15	A
17	A	II-2-8	A	II-4-15	A	II-6-16	A
18	A	II-2-9	A	Divider	A	II-6-17	A
19	A	II-2-10	A	II-5-1	A	II-6-18	A
20	A	II-2-11	A	II-5-2	A	II-6-19	A
21	A	II-2-12	A	II-5-3	A	II-6-20	A
22	A	II-2-13	A	II-5-4	A	II-6-21	A
23	A	II-2-14	A	II-5-5	A	II-6-22	A
24	A	II-2-15	A	II-5-6	B	II-6-23	A
25	A	Divider	A	II-5-7	B	Divider	A
26	B	II-3-1	B	II-5-8	A	II-7-1	A
27	A	II-3-2	A	II-5-9	A	II-7-2	A
28	A	II-3-3	A	II-5-10	A	II-7-3	A
29	A	II-3-4	A	II-5-11	A	II-7-4	A
30	A	II-3-5	A	II-5-12	A	II-7-5	B
Tab	A	II-3-6	A	II-5-13	B	II-7-6	A
Divider	A	II-3-7	A	II-5-14	B	II-7-7	A
I-1	B	II-3-8	A	II-5-15	A	II-7-8	A
I-2	B	II-3-9	A	II-5-16	A	II-7-9	A
I-3	B	II-3-10	A	II-5-17	A	Tab	A
I-4	B	II-3-11	A	II-5-18	A	Divider	A
I-5	B	II-3-12	A	II-5-19	A	III-1-1	A
I-6	B	II-3-13	A	II-5-20	A	III-1-2	A
I-7	B	II-3-14	A	II-5-21	A	Divider	A

PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV
III-2-1	A	IV-2-9	A	V-3-2	A	V-5-6	A
III-2-2	A	IV-2-10	A	V-3-3	A	V-5-7	A
III-2-3	A	Divider	A	V-3-4	A	V-5-8	A
III-2-4	A	IV-3-1	A	V-3-5	A	V-5-9	A
III-2-5	A	IV-3-2	A	V-3-6	A	V-5-10	A
III-2-6	A	IV-3-3	A	V-3-7	A	V-5-11	A
III-2-7	A	IV-3-4	A	V-3-8	A	V-5-12	A
III-2-8	A	IV-3-5	A	V-3-9	A	V-5-13	A
III-2-9	A	IV-3-6	A	V-3-10	A	V-5-14	A
III-2-10	A	Divider	A	V-3-11	A	V-5-15	A
Divider	A	IV-4-1	A	V-3-12	A	V-5-16	A
III-3-1	A	IV-4-2	A	V-3-13	A	V-5-17	A
III-3-2	A	IV-4-3	A	V-3-14	A	V-5-18	A
III-3-3	A	IV-4-4	A	V-3-15	A	V-5-19	A
III-3-4	A	IV-4-5	A	V-3-16	A	V-5-20	A
III-3-5	A	IV-4-6	A	V-3-17	A	V-5-21	A
III-3-6	A	IV-4-7	A	V-3-18	A	V-5-22	A
III-3-7	A	IV-4-8	A	V-3-19	A	V-5-23	A
III-3-8	A	IV-4-9	A	V-3-20	A	V-5-24	A
III-3-9	A	IV-4-10	A	V-3-21	A	V-5-25	A
III-3-10	A	IV-4-11	A	V-3-22	A	V-5-26	A
III-3-11	A	Divider	A	V-3-23	A	V-5-27	A
III-3-12	A	IV-5-1	B	V-3-24	A	V-5-28	A
III-3-13	A	IV-5-2	A	V-3-25	A	V-5-29	A
III-3-14	A	IV-5-3	B	V-3-26	A	V-5-30	A
III-3-15	A	IV-5-4	A	V-3-27	A	V-5-31	A
III-3-16	A	IV-5-5	A	V-3-28	A	Tab	A
III-3-17	A	IV-5-6	A	Divider	A	Divider	A
III-3-18	A	IV-5-7	A	V-4-1	A	A-1	A
III-3-19	A	IV-5-8	B	V-4-2	A	A-2	B
III-3-20	A	IV-5-9	B	V-4-3	A	A-3	B
III-3-21	A	Tab	A	V-4-4	A	A-4	A
III-3-22	A	Divider	A	V-4-5	A	A-5	A
III-3-23	A	V-1-1	A	V-4-6	A	A-6	A
Tab	A	V-1-2	A	V-4-7	A	Divider	A
Divider	A	Divider	A	V-4-8	A	B-1	A
IV-1-1	B	V-2-1	A	V-4-9	A	B-2	A
IV-1-2	B	V-2-2	A	V-4-10	A	B-3	A
IV-1-3	B	V-2-3	A	V-4-11	A	B-4	A
IV-1-4	B	V-2-4	A	V-4-12	A	B-5	A
IV-1-5	B	V-2-5	A	V-4-13	A	B-6	A
Divider	A	V-2-6	A	V-4-14	A	B-7	A
IV-2-1	A	V-2-7	A	V-4-15	A	B-8	A
IV-2-2	A	V-2-8	A	V-4-16	A	B-9	A
IV-2-3	A	V-2-9	A	Divider	A	B-10	A
IV-2-4	A	V-2-10	A	V-5-1	A	B-11	A
IV-2-5	A	V-2-11	A	V-5-2	A	B-12	A
IV-2-6	A	V-2-12	A	V-5-3	A	B-13	A
IV-2-7	A	Divider	A	V-5-4	A	B-14	A
IV-2-8	A	V-3-1	A	V-5-5	A	B-15	A

PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV
B-16	A	B-66	A	E-20	A		
B-17	A	B-67	A	E-21	A		
B-18	A	B-68	A	E-22	A		
B-19	A	B-69	A	E-23	A		
B-20	A	B-70	A	Divider	A		
B-21	A	B-71	A	F-1	B		
B-22	A	B-72	A	F-2	B		
B-23	A	B-73	A	F-3	B		
B-24	A	Divider	A	F-4	B		
B-25	A	C-1	A	F-5	B		
B-26	A	C-2	B	F-6	B		
B-27	A	C-3	B	F-7	B		
B-28	A	C-4	A	Cmt Sht	B		
B-29	A	C-5	A				
B-30	A	C-6	A				
B-31	A	C-7	A				
B-32	A	C-8	A				
B-33	A	C-9	A				
B-34	A	C-10	A				
B-35	A	Divider	A				
B-36	A	D-1	A				
B-37	A	D-2	A				
B-38	A	D-3	A				
B-39	A	D-4	A				
B-40	A	D-5	A				
B-41	A	D-6	A				
B-42	A	D-7	A				
B-43	A	D-8	A				
B-44	A	D-9	A				
B-45	A	D-10	A				
B-46	A	Divider	A				
B-47	A	E-1	A				
B-48	A	E-2	A				
B-49	A	E-3	A				
B-50	A	E-4	A				
B-51	A	E-5	A				
B-52	A	E-6	A				
B-53	A	E-7	A				
B-54	A	E-8	A				
B-55	A	E-9	A				
B-56	A	E-10	A				
B-57	A	E-11	A				
B-58	A	E-12	A				
B-59	A	E-13	A				
B-60	A	E-14	A				
B-61	A	E-15	A				
B-62	A	E-16	A				
B-63	A	E-17	A				
B-64	A	E-18	A				
B-65	A	E-19	A				

THIS PAGE INTENTIONALLY LEFT BLANK.

PREFACE

=====

This manual contains the theory of operation for the CDC® CYBER 180 Model 810 and 830 computer systems. The 810/830 computer system contains four functional units: input/output unit (IOU), central memory (CM), central processing unit (CPU), and maintenance access hardware (MAH). The following is a list of basic equipment and options:

AA161-A	CYBER 180 Models 810/830	BS167-A	2MB (ODPH) Memory Increment
	Basic Equipment	BS222-A	16MB (3HBH) Memory Increment
AT462-A	Model 810 CPU Option	AU125-A	Six Channel Increment
AT463-A	Model 830 CPU Option	GK410-A	Battery Ride Through
AU130-A	CYBER 170 Channel Option	GK212-A	System Power Control Panel
AU129-A	ICI Channel Option	GK419-A	400-Hz Terminator Power
AU127-B	Model 830 Second CPU Option		Supply
AT453-A	Second PP Option		

This manual helps maintenance personnel make more effective use of multilevel block diagrams (MLBDs) to do troubleshooting. Where appropriate, references are made to the MLBD diagram numbers which contain the circuits described. This manual supplements the Multilevel Block Diagrams Hardware Maintenance Manual, publication numbers 60469670 and 60469470, but does not provide one-to-one descriptions of MLBDs.

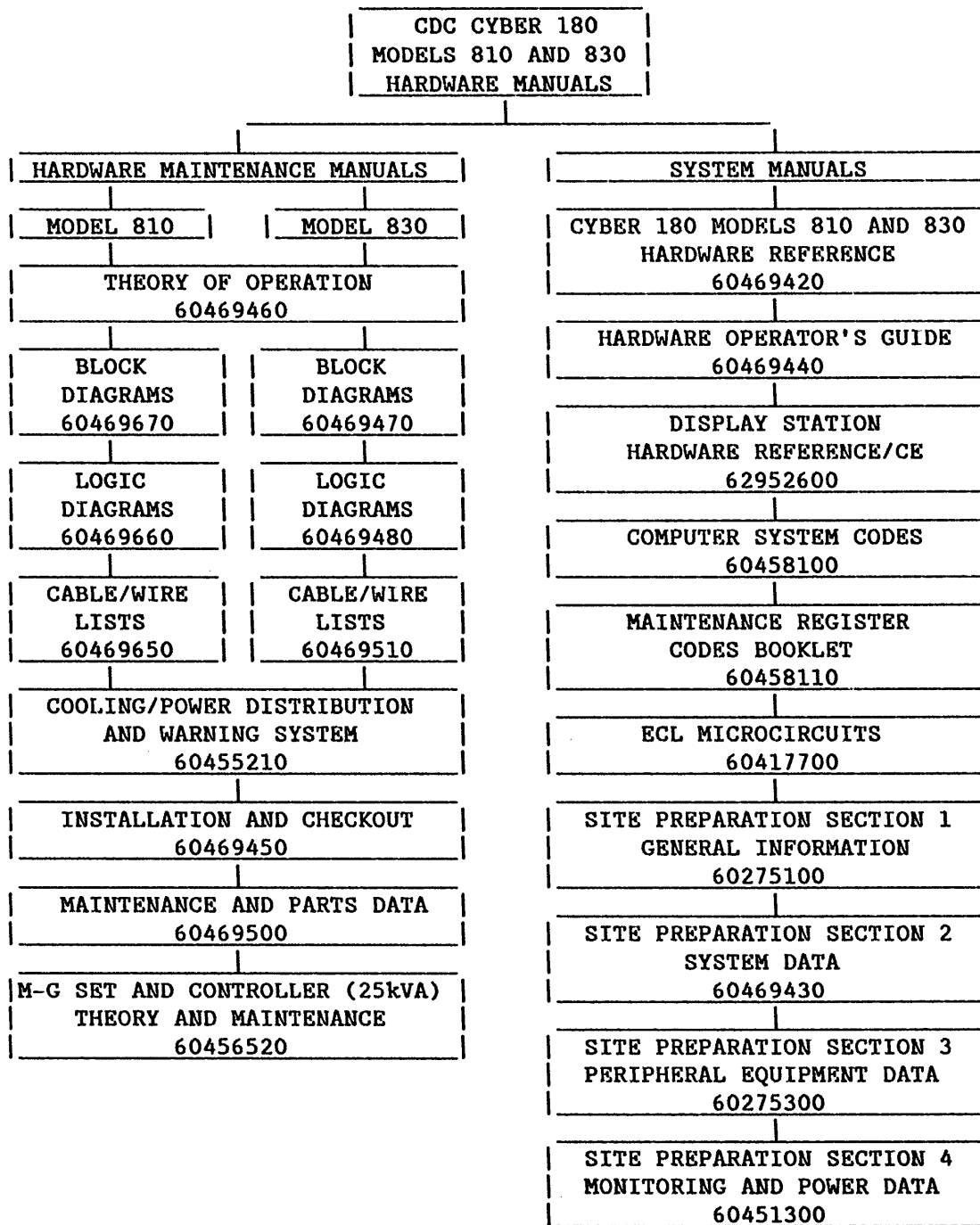
This manual has five parts: an introduction plus one part for each of the four major functional units. Each part is subdivided into sections which correspond to the major functional elements of that unit.

Related manuals are listed in the system publication index on the following page. For other manuals and documents not listed in the system publication index refer to:

<u>Title</u>	<u>Publication Number</u>
MSL 15X Maintenance Software	
Library Reference Manual	60456530
MSL 15X Model Independent Tests	
Maintenance Software Reference Manual	60469390
MSL 151 Maintenance Software	
Reference Manual	60469400

Manual ordering information and latest publication revision levels of nonrestricted manuals are available from the Control Data Literature and Distribution Services catalog, publication number 90310500.

SYSTEM PUBLICATION INDEX



CONTENTS

=====	
PART 1 - INTRODUCTION	I-1
Hardware Components and Configuration	I-1
Central Processing Unit	I-2
Central Memory	I-3
Purpose/Functions	I-3
Assembly/Disassembly Unit	I-4
Ports	I-4
Distributor	I-4
Storage Unit	I-5
Input/Output Unit	I-5
Configuration	I-6
Special Channels	I-6
Standard Equipment	I-6
Options	I-6
 PART II - INPUT/OUTPUT UNIT	
1. GENERAL DESCRIPTION	II-1-1
Configuration	II-1-1
Block Diagram Signal Flow	II-1-3
Major Units	II-1-4
Input/Output Channels	II-1-4
Barrel and Slot	II-1-4
CC545 Display Station Controller	II-1-5
Two Port Multiplexer	II-1-5
IOU Maintenance Registers	II-1-5
Deadstart and Initialization	II-1-6
 2. DEADSTART AND INITIALIZATION	II-2-1
Deadstart Hardware	II-2-1
Microprocessor	II-2-1
Function Decoder 1	II-2-3
Function Decoder 2	II-2-3
Microprocessor ROM	II-2-4
Microprocessor RAM	II-2-4
Parallel Input/Output Port	II-2-6
Universal Asynchronous Receiver/Transmitter	II-2-6
Calendar Clock	II-2-8
CC545 Console	II-2-8
CC555 Terminal	II-2-8
Power-On Master Clear	II-2-9
Master Clear	II-2-9
Two Port Multiplexer	II-2-10

Deadstart Hardwired	II-2-11
Deadstart Software	II-2-12
Long Deadstart Sequence	II-2-12
Short Deadstart Sequence	II-2-13
Diagnostic Switch	II-2-14
 3. BARREL AND SLOT	 II-3-1
A Barrel/Register	II-3-1
A Adder	II-3-2
A Shifter Macrocell	II-3-2
A Rank 1-8 Register Chip 1.2 Macrocell	II-3-3
R Barrel/Register	II-3-3
R+A Adder	II-3-4
R Mux	II-3-4
RA Mux	II-3-4
R Rank 0-7 Register Chips 1-3 Macrocell	II-3-4
Q Barrel/Register	II-3-5
Q Slot Rank 0 Register Chip 1-3 Macrocell	II-3-5
PQ Mux	II-3-5
Q Adder	II-3-5
ADQ-B Mux 1 and 2	II-3-6
ADQ Mux	II-3-6
Q Mux 1 and 2	II-3-6
Q Rank 1-8 Register Chip 1-3 Macrocell	II-3-6
P Barrel/Register	II-3-6
P Slot Rank 0 Register Chip 1-3 Macrocell	II-3-7
ADP-A Mux	II-3-7
P Mux Circuit	II-3-7
PQ Mux	II-3-7
G and GP Mux	II-3-8
P Rank 1-8 Register Chip 1,2 Macrocell	II-3-8
K Barrel/Register	II-3-8
K Barrel Select Mux	II-3-9
K Rank 1-8 Register Chip 1,2 Macrocell	II-3-9
I Register/Barrel	II-3-9
Branch Select Mux	II-3-10
Microcode Control Decode Macrocell	II-3-10
Channel and CM Status Translator	II-3-10
Condition Muxes	II-3-10
I Rank 0-7 Register Chip 1.2 Macrocell	II-3-11
C Barrel/Register	II-3-11
C Input Select Mux	II-3-11
Channel Input/Output Conversion Macrocell	II-3-12
C Register Chip 1.2 Macrocell	II-3-13
Signal Flow	II-3-14
Peripheral Processor Memory	II-3-14
PPM Data Rank 3-6 Macrocell	II-3-15
Y Mux	II-3-15

4. INPUT/OUTPUT CHANNELS	II-4-1
Configuration	II-4-2
CYBER 170 Channels	II-4-2
Internal Interface	II-4-2
Full (CHL 2.0)	II-4-3
Active (CHL 2.0)	II-4-4
Flag (CHL 2.0)	II-4-4
Error Flag (CHL 2.0)	II-4-4
External Interface	II-4-4
Channel Protocol	II-4-5
Data Output Sequence	II-4-5
Data Input Sequence	II-4-7
ICI Channels	II-4-8
Internal Interface	II-4-9
External Interface	II-4-9
Control Signals	II-4-9
Master Clear	II-4-9
Data/Parity	II-4-9
ICI Timing	II-4-9
Data/Parity Transmission Scheme	II-4-10
Channel Control Macrocell	II-4-12
Channel Data Macrocell	II-4-12
Signal Definitions	II-4-12
Channel 0 External Data, Channel 1 Received Data	II-4-13
Barrel 0, 1 Channel Data	II-4-13
Barrel 0, 1 Channel 0/1/2 Function or Full	II-4-13
Channel 0-2 Select Barrel 0, 1 Rank 9	II-4-13
Master Clear Delay	II-4-13
Channel 0, 1 Load	II-4-13
Barrel 0, 1 Channel 0/1/2 Empty	II-4-14
Channel 1 External P2	II-4-14
Channel 0-2 P3	II-4-14
Channel 0, 1 Data	II-4-14
Channel 0/1/2 Barrel 0, 1 Data	II-4-14
Data I/O Sequences	II-4-14
MAC Interface Shifter	II-4-15
5. IOU MAINTENANCE REGISTERS	II-5-1
Maintenance Registers	II-5-1
IOU Maintenance Registers to MAC Interfaces	II-5-3
Maintenance Register Functions	II-5-3
Read	II-5-3
Write	II-5-4
Master Clear Assembly Disassembly Unit (ADU)	II-5-4
Clear Fault Status Register	II-5-4
Request Summary Status Byte	II-5-4
Maintenance Register Bit Description	II-5-5
Summary Status Byte Register (Read only)	II-5-5
Element ID Register (Read Only)	II-5-8
Options Installed Register (Read Only)	II-5-8

Fault Status Mask Register (Read/Write)	II-5-8
Operating System Bounds Register (Read/Write)	II-5-9
Environment Control Register (Read/Write)	II-5-12
Status Register (Read Only)	II-5-14
Fault Status 1 Register (Read/Write)	II-5-15
Fault Status 2 Register (Read/Write)	II-5-19
Test Mode Register (Read/Write)	II-5-20
 6. TWO PORT MULTIPLEXER	 II-6-1
General Description	II-6-1
External Interface	II-6-3
RS-232-C Interface	II-6-3
Baud Rate Switch	II-6-3
Port Options Switch	II-6-3
RS-232-C Signals	II-6-5
External Device to PP Data Flow	II-6-5
RS-366 Interface	II-6-6
Call Request	II-6-7
Digit Present	II-6-7
Four Data Bits	II-6-7
Power Indication	II-6-8
Data Line Occupied	II-6-8
Present Next Digit	II-6-8
Abandon Call	II-6-8
Call Origination Status	II-6-8
PP to External Device Data Flow	II-6-8
Power Control Interface	II-6-10
Calendar Clock	II-6-10
Power Control	II-6-10
Internal Interface	II-6-10
PP to TPM Function Codes	II-6-10
Programming Considerations	II-6-18
Request To Send and Data Terminal Ready	II-6-18
External Device to TPM Functions	II-6-20
Remote Deadstart	II-6-20
Auto Answer	II-6-20
Remote Power Control	II-6-20
 7. CC545 DISPLAY STATION CONTROLLER	 II-7-1
Interface Definition	II-7-1
DSC to PP	II-7-1
DSC to Console	II-7-1
DSC to Microprocessor	II-7-2
Operating Modes	II-7-2
12-Bit Mode Operation	II-7-2
Character Mode	II-7-4
Dot Mode	II-7-6
Keyboard Input Mode	II-7-6
16-Bit ASCII Mode Operation	II-7-6

Character Mode	II-7-7
Dot Mode	II-7-9
Keyboard Input Mode	II-7-9
Performance Characteristics	II-7-9

PART III - MAINTENANCE ACCESS HARDWARE

1. INTRODUCTION	III-1-1
2. UNIT DESCRIPTION	III-2-1
Maintenance Control Unit (MCU)	III-2-1
Maintenance Channel (MCH)	III-2-1
Time Out	III-2-2
MCH Interface to PP	III-2-2
MCH Interface to MAC (Radial Interfaces)	III-2-2
Maintenance Access Control (MAC)	III-2-5
CPU Interface	III-2-6
Maintenance Register Interfaces	III-2-7
3. OPERATIONS	III-3-1
MCU and MCH Operation	III-3-1
MCU Coding Examples	III-3-2
MAC Operation	III-3-4
Overall Operation	III-3-4
MAC Circuits	III-3-5
Parity and Error Reporting	III-3-5
Access Control (MAC 2.2)	III-3-5
Priority (MAC 2.7)	III-3-5
Bit Serial Control (MAC 2.6)	III-3-6
Nanocode Organization	III-3-9
Nanocode Format	III-3-9
Nanocode Map	III-3-11
Nanocode Sequences	III-3-11
Entry Points	III-3-11
Addressing the Nanocrands (MAC 2.3)	III-3-12
MAC Operation without Nanocode	III-3-15
MAC Operation with Nanocode	III-3-15
MAC Requests	III-3-16
CPU	III-3-17
MAC Sequences--No Microcode Required	III-3-17
Full Word Transfers	III-3-17
Partial Word Transfers	III-3-18
MAC Sequences--Microcode Required	III-3-20
CPU Processor Requests	III-3-21
Full Word Transfers	III-3-22
Partial Word Transfers	III-3-22

PART IV - CENTRAL MEMORY

1. INTRODUCTION	IV-1-1
Description	IV-1-1
Assembly/Disassembly Unit (ADU)	IV-1-4
Unit Description	IV-1-4
ADU Priority	IV-1-4
Ports	IV-1-4
Distributor	IV-1-4
Storage Unit	IV-1-4
CM Functions	IV-1-5
CM Features	IV-1-5
2. ASSEMBLY DISASSEMBLY UNIT	IV-2-1
ADU Instruction	IV-2-1
ADU Instruction Format	IV-2-1
Configuration	IV-2-2
CM Addressing	IV-2-2
ADU Data Transfers/Manipulations	IV-2-3
60-Bit Disassembly (CM to PPM)	IV-2-3
Central Read from (A) to d (0060 d)	IV-2-3
Central Read (d) Words from (A) to m (0061 d m)	IV-2-4
64-Bit Disassembly (CM to PPM)	IV-2-5
Central Read from (A) to d Long (1060 d)	IV-2-5
Central Read (d) Words from (A) to m Long (1061 d m)	IV-2-5
60-Bit Assembly (PPM to CM)	IV-2-6
Central Write from d to (A) (0062 d)	IV-2-6
Central Write (d) Words from m to (A) (0063 d m)	IV-2-6
64-Bit Assembly (PPM to CM)	IV-2-7
Central Write from d to (A) Long (1062 d)	IV-2-7
Central Write (d) Words from m to (A) Long (1063 d m)	IV-2-7
12-Bit/16-Bit Word	IV-2-8
ADU Major Circuit Components	IV-2-8
ADU Assembly Mux	IV-2-8
ADU Read/Write Buffer	IV-2-8
Data To ADU Disassembly Mux	IV-2-8
3. PORT INTERFACES	IV-3-1
CPU/CPU Port Interface	IV-3-1
Port Input Lines	IV-3-2
Port Output Lines	IV-3-3
IOU/IOU ADU Port Interface	IV-3-4
Port Input Lines	IV-3-5
Port Output Lines	IV-3-5
4. DISTRIBUTOR	IV-4-1
Port Priority Network	IV-4-1
Memory Functions	IV-4-3

Memory Responses	IV-4-5
Memory Operations	IV-4-6
Refresh Operation	IV-4-6
Read Operation	IV-4-6
Write Operation	IV-4-7
Read/Modify/Write (Partial Write) Operation	IV-4-8
Read and Set Lock	IV-4-8
Read and Clear Lock	IV-4-9
Exchange	IV-4-9
CYBER 170 Exchange Request	IV-4-9
Read Exchange Address	IV-4-10
Read Free Running Counter	IV-4-10
Refresh Counter Resync	IV-4-10
Interrupt	IV-4-11

5. STORAGE UNIT IV-5-1

Memory Arrays and Addressing Scheme	IV-5-1
Address Wrap-Around	IV-5-1
CM Reliability, Accessibility, and Maintainability Features	IV-5-4
Parity	IV-5-4
Maintenance Registers	IV-5-4
SECDED	IV-5-4
Bounds Register	IV-5-5
Memory Configuration Switch Register	IV-5-8

PART V - CENTRAL PROCESSING UNIT

1. INTRODUCTION V-1-1

2. CONTROL STORE V-2-1

Hardware Description	V-2-1
Writable Control Store	V-2-2
Control Store Word	V-2-2
Control Store Organization	V-2-2
Control Store Timing	V-2-2
Micrand Holding Register (MHR)	V-2-3
Control Store Address Register (S Register)	V-2-3
Control Store Address Multiplexer (S Mux)	V-2-3
Return Register 1 (R1 Reg), Return Register 2 (R2 Reg)	V-2-4
Micrand Address Selection Logic (MAS)	V-2-4
Page Offset Adder (R Adder)	V-2-4
R Multiplexer (R Mux)	V-2-4
Model Dependent CPU Registers	V-2-5
Hardware Operations	V-2-5
Control Store Addressing	V-2-5
MAS Field	V-2-6
BCOND Field	V-2-7
Loading Control Store	V-2-8
Start/Stop Control Store	V-2-9

Normal Sequence	V-2-9
Deadstart	V-2-9
Instruction Exit	V-2-9
Breakpoint	V-2-10
Sweep Mode	V-2-10
Microstep	V-2-10
Instruction Step	V-2-10
Maintenance Functions	V-2-10
Parity	V-2-10
Read Control Store via MCH	V-2-11
Maintenance Scan	V-2-11
Control Store Error Log	V-2-12
 3. EXECUTION UNITS	 V-3-1
Execution Units Descriptions	V-3-1
64-Bit Unit	V-3-2
ASX Mux (Register File A Data Sign Extended Mux)	V-3-2
SBD Mux (Register File B Data Mux)	V-3-2
B adder	V-3-3
Shifter	V-3-3
Execution Sense Field (ESC)	V-3-3
18-Bit Unit	V-3-4
Exponent Unpackers	V-3-4
18-Bit Adder	V-3-4
Normalizer	V-3-5
18-Bit Latch	V-3-5
Shift Count Select Mux	V-3-5
Exponent Packer	V-3-6
Byte Unit	V-3-6
AD/BD Disassembly	V-3-7
Writing the Register File from XBD	V-3-7
Data ROM and Mux (Preprocess Path)	V-3-7
Byte Scan Mux	V-3-7
Decimal Operations	V-3-7
Add/Subtract	V-3-7
Binary to Decimal Convert	V-3-8
XAO Mux, CHC ROM, and E ROM (Post-Process Path)	V-3-8
Compare Operations	V-3-8
Decimal Compare	V-3-8
Byte Compare	V-3-8
Stream Unit	V-3-8
Load Multiple Registers from CM	V-3-9
Store Multiple Registers in CM	V-3-9
Process Multiple Bytes for the BDP	V-3-9
Execution Unit Interface Descriptions	V-3-10
Register File Unit	V-3-10
Reading and Writing RF	V-3-10
Micrand Fields RGA, RGB, RGC	V-3-12
Microtrip Mechanism	V-3-12
Catchable Conditions	V-3-12
Non-Catchable Conditions	V-3-13
Microtraps MCR/UCR	V-3-13

Monitor Condition Register	V-3-13
User Condition Register	V-3-15
Trap, Exchange, and Halt Microtraps	V-3-16
Trap Microtrap	V-3-16
Exchange Microtrap	V-3-16
Halt Microtrap	V-3-16
Timing	V-3-16
Debug Microtrap Operation	V-3-17
Processor Fault Status (PFS) Error Microtrap	V-3-18
Floating Point Microtrap	V-3-18
Map No Hit Microtrap	V-3-18
CYBER Address Out of Range Microtrap (60-Bit Mode Only)	V-3-18
Miscellaneous Execution Control	V-3-18
Retry Enable	V-3-18
Retry-Related Flip-Flop	V-3-19
Retry in Progress	V-3-19
Parity Trap in Progress	V-3-19
Retry Enable Flip-Flop	V-3-19
Execution Unit Advance Pipe	V-3-19
Primitive Full (PFF)	V-3-19
Exit Condition Met	V-3-19
Parity	V-3-20
Primitive Formation Unit	V-3-20
Execution Unit Operations	V-3-21
Multiply/Divide Operation	V-3-21
Multiply (General Information)	V-3-21
Multiply Algorithm	V-3-21
Number of Iterations	V-3-22
Basic Micrand Flow	V-3-24
Divide Operation (General Information)	V-3-24
Number of Iterations	V-3-24
General Divide Algorithm	V-3-24
Floating Point Exceptions	V-3-25
Early Exceptions Canned Results	V-3-25
Debug Operation	V-3-25
Debug Microtrap	V-3-27
 4. READ NEXT INSTRUCTION	 V-4-1
Hardware Overview	V-4-1
P0 Register	V-4-1
P1 Register	V-4-1
I Multiplexer (I Mux)	V-4-2
F Latch	V-4-3
RNI Address Register/Incrementer	V-4-3
P Register/Incrementer	V-4-4
Hardware Functions	V-4-5
RNI Initialization	V-4-5
Steady State RNI	V-4-6
RNI Words and Addressing	V-4-6
Instruction Decode for Length	V-4-6
BDP Instruction Lengths	V-4-8

MAP Faults	V-4-8
Debug Pretrap	V-4-9
Halting the RNI Unit	V-4-9
Initialization for Stream Requests	V-4-9
Steady-State Stream Operation	V-4-11
Resetting RNI Operation after Streaming	V-4-11
Terminating Stream Operation	V-4-11
60-Bit Mode Operation	V-4-12
60-Bit Mode Shift	V-4-12
60-Bit Mode Length Decodes	V-4-12
Parcel Boundary Faults	V-4-14
Word Boundary Checking	V-4-14
Maintenance Functions	V-4-14
Read P Register	V-4-14
Read RNI Register	V-4-14
Performance Monitoring Features	V-4-14
Ident Codes	V-4-15
RNI Requests	V-4-15
Branch Requests	V-4-15
Stream Request	V-4-15
 5. MAP	 V-5-1
Unit Descriptions	V-5-1
Map Input Register	V-5-1
FLC Register	V-5-1
Page Size Mask Register	V-5-1
Associative Files	V-5-4
RMA Register	V-5-4
Real Files	V-5-6
Validity Files	V-5-7
UTP Register	V-5-7
MAP Sense Mux	V-5-8
Hardware Functions	V-5-9
Hardware Sequences	V-5-9
Hardware Cycles	V-5-9
Port Busy	V-5-9
RMA Path	V-5-9
UTP Register Strobing	V-5-10
Validity Checking	V-5-10
Fault Reporting	V-5-10
Functions Initiated by Execution	V-5-10
Functions Initiated by RNI	V-5-11
Translate with No Validation (Function Code 13 Hex)	V-5-11
RNI-Initiated Stream Request	V-5-11
MAP Lockup	V-5-11
Loading the Files	V-5-12
Central Memory Function Recode	V-5-13
P Register Key Modifications	V-5-14
Call/Branch	V-5-14
Return	V-5-15
Jump (mnemonic IBRANCH JUMP, code = 3)	V-5-16

Branch (mnemonic IBRANCH INST, code = 4)	V-5-16
Call (mnemonic IBRANCH CALL, code = 5)	V-5-17
Return (mnemonic IBRANCH RET, code = 6)	V-5-17
Exchange (mnemonic IBRANCH EXCH, code = 7)	V-5-17
P Ring Modifications	V-5-17
Branch (mnemonic IBRANCH INST)	V-5-17
Call (mnemonic IBRANCH CALL)	V-5-17
Return/Exchange (mnemonic IBRANCH RET/EXCH)	V-5-17
Modified Bit	V-5-17
LRU	V-5-18
Operation with All Files Enabled	V-5-19
Operation With Files Disabled	V-5-19
Monitor and Job Entries	V-5-19
MAP Functions	V-5-20
MAP Sense	V-5-24
Key Field Sense (Bits 0-15)	V-5-26
RN, SEG Field Sense (Bits 16-31)	V-5-26
BN Field Sense (Bits 32-63)	V-5-26
Branch Conditions	V-5-28
Exit Condition	V-5-28
60-Bit Mode Features	V-5-28
Maintenance Features	V-5-29
RMA Mode	V-5-30
UX	V-5-30
GA	V-5-30
MAP Kill	V-5-31

APPENDIXES

A. GLOSSARY OF TERMS	A-1
B. IOU INSTRUCTION SUMMARY	B-1
Instruction Set	B-1
Instruction Formats	B-1
12-Bit Versus 16-Bit Exceptions	B-2
Address Modes	B-2
No-Address Mode	B-2
Constant Mode	B-2
Direct Mode	B-2
Indirect Mode	B-2
Memory Mode	B-3
Address Mode Instructions	B-3
Short Word (12-Bit) Stores	B-10
Use of Location 0	B-10
Use of Location 7777	B-10
Memory address mode instructions (m+(d))	B-10
Unused Bits	B-11
IOU Flowcharts	B-11
IOU Instruction Timing	B-42

B-46

B-46

B-46
B-46
B-46
B-46
B-46
B-47
B-47
B-47
B-47
B-47
B-47
B-47
B-47
B-48
B-48

B-48

B-48
B-48
B-49
B-49
B-49
B-49
B-49
B-50
B-50
B-50
B-50
B-50
B-51
B-51

B-51

B-51
B-52
B-52
B-52
B-52
B-53
B-53
B-53
B-53
B-53
B-54
B-54
B-54
B-54

B-54

B-55
B-55
B-55

Add One (d) Long	1036 d [AODL d]	B-55
Subtract One (d)	0037 d [SOD d]	B-55
Subtract One (d) Long	1037 d [SODL d]	B-56
Replace Add ((d))	0045 d [RAI d]	B-56
Replace Add ((d)) Long	1045 d [RAIL d]	B-56
Add One ((d))	0046 d [AOI d]	B-56
Add One ((d)) Long	1046 d [AOIL d]	B-57
Subtract One ((d))	0047 d [SOI d]	B-57
Subtract One ((d)) Long	1047 d [SOIL d]	B-57
Replace Add (m+(d))	0055 d m [RAM m,d]	B-57
Replace Add (m+(d)) Long	1055 d m [RAML m,d]	B-58
Add One (m+(d))	0056 d m [AOM m,d]	B-58
Add One (m+(d)) Long	1056 d m [AOML m,d]	B-58
Subtract One (m+(d))	0057 d m [SOM m,d]	B-59
Subtract One (m+(d)) Long	1057 d m [SOML m,d]	B-59
Branch		B-59
Unconditional Jump d	0003 d [UJN d]	B-59
Zero Jump d	0004 d [ZJN d]	B-59
Non-Zero Jump d	0005 d [NJN d]	B-60
Plus Jump d	0006 d [PJN d]	B-60
Minus Jump d	0007 d [MJN d]	B-60
Long Jump to m+(d)	0001 d m [LJM m,d]	B-60
Return Jump to m+(d)	0002 d m [RJM m,d]	B-60
ADU Instructions		B-60
Load R Register	0024 d [LRN d]	B-61
Store R Register	0025 d [SRD d]	B-61
Central Read from (A) to d	0060 d [CRD d]	B-61
Central Read from (A) to d Long	1060 d [CRDL d]	B-61
Central Read (d) Words From (A) to m	0061 d m [CRM m,d]	B-62
Central Read (d) Words From (A) to m Long	1061 d m [CRML m,d]	B-62
Central Write From d to (A)	0062 d [CWD d]	B-63
Central Write From d to (A) Long	1062 d [CWDL d]	B-63
Central Write (d) Words From m to (A)	0063 d m [CWM m,d]	B-64
Central Write (d) Words From m to (A) Long	1063 d m [CWML m,d]	B-64
Central Read and Set Lock From d to (A)	1000 d [RDSL d]	B-65
Central Read and Clear Lock From d to (A)	1001 d [RDCL d]	B-65
Input/Output		B-65
Jump to m if Channel c Active	00640 c m [AJM m,c]	B-66
Test and Set Channel c Flag	00641 c m [SCF m,40B+c]	B-66
Jump to m if Channel c Flag Set	1064X c m [FSJM m,c]	B-66
Jump to m if Channel c Inactive	00650 c m [IJM m,c]	B-66
Clear Channel c Flag	00651 c m [CCF 40B+c]	B-66
Jump to m if Channel c Flag Clear	1065X c m [FCJM m,c]	B-66
Jump to m if Channel c Full	00660 c m [FJM m,c]	B-67
Test and Clear Channel c Error Flag When Set	00661 c m [SFM m,40B+c]	B-67
Jump to m if Channel c Empty	00670 c m [EJM m,c]	B-67
Test and Clear Channel c Error Flag When Clear	00671 c m [CFM m,40B+c]	B-67
Input to A From Channel c When Active and Full	00700 c [IAN c]	B-67
Input to A From Channel c if Active and Full	00701 c [IAN 40B+c]	B-67
Input (A) Words to m From Channel c	0071X c m [IAM m,c]	B-68

Input (A) Words to m From Channel c Packed 1071X c m [IAPM m,c]	B-68
Output From A to Channel c 00720 c [OAN c]	B-69
Output From A to Channel c 00721 c [OAN 40B+c]	B-69
Output (A) Words From m to Channel c 0073X c m [OAM m,c]	B-70
Output (A) Words From m to Channel c Packed 1073X c m [OAPM m,c]	B-70
Activate Channel c When Inactive 00740 c [ACN c]	B-70
Unconditionally Activate Channel c 00741 c [ACN 40B+c]	B-71
Deactivate Channel c When Active 00750 c [DCN c]	B-71
Unconditionally Deactivate Channel c 00751 c [DCN 40B+c]	B-71
Function (A) on Channel c When Inactive 00760 c [FAN c]	B-71
Function (A) on Channel c if Inactive 00761 c [FAN 40B+c]	B-71
Function m on Channel c When Inactive 00770 c m [FNC m,c]	B-72
Function m on Channel c if Inactive 00771 c m [FNC m,40B+c]	B-72
Other Instructions	B-72
Pass 0000 d [PSN] 0024 0 0025 0 0027 x	B-72
PP Keypoint 0027 [KEYP]	B-72
Exchange Jump 0026 d	B-73
Interrupt Processor 1026 d [INPN d]	B-73

C. IOU MICRAND FIELDS C-1

Micrand Bit Assignment	C-6
A Adder Code Bits 0-4 (Micrand Bits 8 through 12)	C-7
Q Adder Code Bits 0, 1 (Micrand Bits 13 and 14)	C-7
A Select Condition (Micrand Bits 24 through 26)	C-7
Q Select Condition (Micrand Bits 27 through 29)	C-8
P Select Condition (Micrand Bits 30 through 33)	C-8
G Select Condition (Micrand Bits 34 through 37)	C-9
Exit Condition (Micrand Bits 38 through 40)	C-9
PP Write Condition (Micrand Bits 41 through 43)	C-9
Branch Condition (Micrand Bits 44 through 48)	C-10
Y Mux Select - Branch 1 on BAS 2.7 (Micrand Bits 59 and 60)	C-10
ADU/Channel Controls (Word Count/Channel Code)	C-11
ADU/Channel Functions (Micrand Bits 79 through 83)	C-12

D. MAC/MICROCODE AIDS D-1

Aids for Understanding MAC/Microcode Operations	D-1
---	-----

E. CPU MICRAND FIELDS E-1

Write Mux Select (WMX)	E-1
Register File A, B, C - RGA, RGB, RGC	E-2
mmmm	E-4
YYYY	E-4
BBBB	E-6
CCCC	E-6
AD Multiplexer Selects (ADS)	E-6
BD Multiplexer Selects (BDS)	E-7

Register Partial Writes (RFPW)	E-7
Shifter Control (SHC)	E-8
Shift Count Select (SCS)	E-9
Unpacker Control (FP)	E-9
S Adder Latch Control (Latch)	E-10
Big Adder Control (B Adder)	E-10
Small Adder (S Adder) Control	E-11
S Adder Function (Format A+E)	E-11
Execution Sense Field (ESC)	E-11
S Adder ESC (Format A+E)	E-11
B Adder ESC (Format B)	E-11
Normalization (Norm)	E-13
ER Multiplexer (ER Mux)	E-13
J3, K3 and C3 Control	E-13
Mux Select (JCNT)	E-13
K Select (KCNT)	E-14
C Mux Select (CCNT)	E-14
YKW Mux Select (YKW)	E-14
L Mux Select (L MUX)	E-14
Counter CO (LENJ, LENK, LENC)	E-15
Multiply Subfunction Bits	E-15
Mux Select and Function Code (XBD)	E-15
Data Mux and Overflow Control (DMX)	E-16
Data Mux Control	E-16
Overflow Control	E-16
E Mode FF Control and Slack Digit (EB)	E-17
Source Sign Control (GSN)	E-17
J Sign Record (JR)	E-17
K Sign Record (KR)	E-17
Source Field Fill Control (FILL)	E-18
Q Count Control (Q, QC)	E-18
XAO Mux Control (XAO)	E-18
Byte Sense Conditions (SENZ)	E-19
BDP Exit Control (XBDP)	E-19
Invert Control (PAS2)	E-20
J Sign Fix (JFX)	E-20
K Sign Fix (KFX)	E-20
Multiply/Divide Subfunction (SUBF)	E-20
Floating Point Function (FP FUNC)	E-21
Floating Point Trap Allow (TRP)	E-21
Floating Point Exceptions	E-22
Early Exception	E-22
Floating Point Microtrap	E-22
Late Trap	E-23
Ring-Ring Selection	E-23
Exit	E-23
 F. INSTRUCTION EXECUTION TIMES	 F-1
64-Bit Mode Instruction Timings	F-1

FIGURES

I-1	System Configuration	I-1
I-2	Central Processor Diagram	I-2
I-3	Central Memory Diagram	I-4
I-4	IOU Diagram	I-5
II-1-1	IOU Block Diagram	II-1-2
II-2-1	Deadstart Microprocessor	II-2-2
II-2-2	Deadstart Function Decoders	II-2-4
II-2-3	Deadstart ROM, RAM, and PIO Ports 0, 1, and 2	II-2-5
II-2-4	Universal Asynchronous Receiver/Transmitters	II-2-7
II-2-5	Two Port Mux	II-2-10
II-4-1	Data Output Sequence	II-4-6
II-4-2	Data Input Sequence	II-4-8
II-4-3	Integrated Controller Interface (ICI)	II-4-10
II-4-4	ICI Timing	II-4-11
II-5-1	IOU Maintenance Registers	II-5-2
II-5-2	IOU Maintenance Register Bit Assignment	II-5-6
II-5-3	Operating System Bounds Comparison	II-5-11
II-6-1	Two Port Mux	II-6-2
II-6-2	RS-232-C Protocol	II-6-5
III-1-1	Maintenance Access Hardware Organization	III-1-2
III-3-1	Bit-Serial Interface	III-3-7
III-3-2	MAC Serial Timing	III-3-8
III-3-3	Flow Diagram Format	III-3-17
III-3-4	Full Word Transfers Flow Diagram	III-3-18
III-3-5	Partial Word Transfers Flow Diagram	III-3-19
III-3-6	Microcode Request	III-3-20
III-3-7	Full Word Transfer--Microcode Required	III-3-21
III-3-8	Partial Word Transfer, Microcode Required	III-3-23
IV-1-1	CM Organization	IV-1-2
IV-1-2	Central Memory Block Diagram	IV-1-3
IV-3-1	CM Ports	IV-3-1
IV-3-2	CPU/CPU Port Interface	IV-3-2
IV-3-3	IOU/IOU ADU Port Interface	IV-3-4
IV-4-1	CM Distributor	IV-4-2
IV-5-1	CM Storage Unit	IV-5-2
IV-5-2	Memory Array Addressing Scheme (Interleaved Mode)	IV-5-3
IV-5-3	256K-Chip Memory Array Addressing Scheme (Interleaved Mode)	IV-5-4
V-1-1	Computer System Models 810 and 830 Block Diagram	V-1-2

V-2-1	Control Store	V-2-1
V-2-2	Control Store Word	V-2-2
V-2-3	Micrand Address Select	V-2-6
V-3-1	Relationship between S, MHR, Primitive, and Write Times of a Micrand	V-3-2
V-3-2	64-Bit Adder Circuit	V-3-3
V-3-3	18-Bit Adder Circuit	V-3-4
V-3-4	Unpacker Circuits	V-3-5
V-3-5	Exponent Packer Circuit	V-3-6
V-3-6	Register File Allocation	V-3-11
V-3-7	Scan Technique of the Multiplier	V-3-22
V-3-8	Example of Binary Multiply Using a Multiplier Recoding Algorithm	V-3-23
V-4-1	Instruction Parcelling	V-4-7
V-4-2	Halting the RNI Unit	V-4-10
V-4-3	60-Bit Mode Shift	V-4-13
V-4-4	Ident Tags	V-4-16
V-5-1	MAP	V-5-2
V-5-2	Formation of Page Number and Page Offset	V-5-3
V-5-3	Associative Files	V-5-5
V-5-4	Real File	V-5-6
V-5-5	Validity Files and SDE	V-5-8
V-5-6	Branch/Call Key Modifications	V-5-14
V-5-7	Return Key Modifications	V-5-15
V-5-8	Call Ring Modifications	V-5-18
V-5-10	MAP Sense	V-5-25
V-5-11	RN, SEG Field Sense Decode	V-5-27
B-1	No Address Mode Example	B-4
B-2	Constant Mode Example	B-4
B-3	Direct Mode Example, 12- and 16-bit Instructions	B-5
B-4	Direct Mode Example, Long Jump Instruction	B-6
B-5	Direct Mode Example, Return Jump Instruction	B-7
B-6	Indirect Mode Example	B-8
B-7	Memory Mode Example	B-9
B-8	Flowchart, PP Instruction 00	B-11
B-9	Flowchart, PP Instructions 01 and 02	B-12
B-10	Flowchart, PP Instructions 03 through 07	B-13
B-11	Flowchart, PP Instructions 10 through 17	B-14
B-12	Flowchart, PP Instructions 20 through 23	B-15
B-13	Flowchart, PP Instructions 24 and 25	B-16
B-14	Flowchart, PP Instructions 122 through 124	B-17
B-15	Flowchart, PP Instruction 26	B-18
B-16	Flowchart, PP Instruction 126	B-19
B-17	Flowchart, PP Instructions 30 through 37, 130 through 137	B-20
B-18	Flowchart, PP Instructions 40 through 47, 140 through 147	B-21
B-19	Flowchart, PP Instructions 50 through 57, 150 through 157	B-22
B-20	Flowchart, PP Instruction 60	B-23
B-21	Flowchart, PP Instruction 160	B-24
B-22	Flowchart, PP Instruction 61	B-25

B-23	Flowchart, PP Instruction 161	B-26
B-24	Flowchart, PP Instruction 62	B-27
B-25	Flowchart, PP Instruction 162	B-28
B-26	Flowchart, PP Instruction 63	B-29
B-27	Flowchart, PP Instruction 163	B-30
B-28	Flowchart, PP Instructions 64 through 67	B-31
B-29	Flowchart, PP Instructions 164 and 165	B-32
B-30	Flowchart, PP Instructions 70 and 72	B-33
B-31	Flowchart, PP Instruction 71	B-34
B-32	Flowchart, PP Instruction 171	B-35
B-33	Flowchart, PP Instruction 73	B-36
B-34	Flowchart, PP Instruction 173	B-37
B-35	Flowchart, PP Instructions 74 and 75	B-38
B-36	Flowchart, PP Instructions 76 and 77	B-39
B-37	Flowchart, PP Instruction 100	B-40
B-38	Flowchart, PP Instruction 101	B-41
D-1	Excerpt from DQ Pak	D-7
D-1	MAC Internal Interface	D-8
D-3	MAC Paths to/from Maintenance Registers	D-9

TABLES

II-3-1	A Adder Code	II-3-2
II-3-2	ADQ-B Mux 1 and 2 Selection	II-3-6
II-3-3	P Mux Circuit Selection	II-3-7
II-3-4	PQ Mux Selection	II-3-8
II-3-5	G Data Selection	II-3-8
II-3-6	K Barrel Select Mux Selection	II-3-9
II-3-7	Branch Select Mux Selection	II-3-10
II-3-8	C Input Select Mux Selection	II-3-11
II-3-9	Conversion Muxes and Parity Mux Selection	II-3-12
II-4-1	Channel Allocation	II-4-3
II-4-2	RWCH Bits 73 through 75	II-4-7
II-4-3	Example of Data I/O Sequences	II-4-15
II-5-1	IOU Maintenance Registers	II-5-1
II-5-2	Summary Status Byte Register	II-5-5
II-5-3	Element ID Register	II-5-8
II-5-4	Options Installed Register	II-5-9
II-5-5	Fault Status Mask Register	II-5-10
II-5-6	Operating System Bounds Register	II-5-11
II-5-7	Environment Control Register	II-5-13
II-5-8	Status Register	II-5-15
II-5-9	Fault Status 1 Register	II-5-16
II-5-10	Fault Status 2 Register	II-5-19
II-5-11	Test Mode Register	II-5-20
II-5-12	Test Mode Function	II-5-21

II-6-1	Baud Rate Selection Switch	II-6-4
II-6-2	Port Options Switch	II-6-4
II-6-3	Digit Code of RS-366 Data Bits	II-6-7
II-6-4	PP to TPM Function Codes	II-6-11
II-6-5	TPM Read Calendar Clock	II-6-14
II-6-6	Write Calendar Clock	II-6-14
II-6-7	Output Data Procedure	II-6-19
II-6-8	Input Data Procedure	II-6-19
II-7-1	DSC-Console Signals	II-7-2
II-7-2	DSC-Microprocessor Signals	II-7-3
II-7-3	12-Bit Function Codes	II-7-3
II-7-4	Character Generation Codes	II-7-5
II-7-5	16-Bit Function Codes	II-7-7
II-7-6	ASCII Display in the Console	II-7-8
II-7-7	Keyboard Input Characters and ASCII Code	II-7-9
III-2-1	Decoding of Connect Code	III-2-2
III-2-2	Signals between MCH and RI/MAC	III-2-3
III-2-3	Op Code Functions and Type Code Assignments	III-2-6
III-2-4	IOU Register Read/Write via MAC	III-2-8
III-2-5	CM Registers Read/Write via MAC	III-2-8
III-2-6	CPU Registers Read/Write via MAC	III-2-9
III-3-1	IOU/CM Register Byte for Pak Select 0 through 4	III-3-9
III-3-2	Register Byte for Pak Sel 5 through 7, E and F	III-3-10
III-3-3	Control Word Mapping ROM	III-3-13
III-3-4	Nonrelocatable Starting Addresses	III-3-14
IV-1-1	Memory Sizes and Equipment Requirements	IV-1-1
IV-2-1	CM Address Formation	IV-2-3
IV-2-2	12/16 Bit Assembly/Disassembly	IV-2-9
IV-2-3	Pak Address Bits and Pak Write Enable Assignment	IV-2-10
IV-4-1	Request Lockout Time in Bank Cycles	IV-4-1
IV-4-2	Action for a Given Memory Function Failure	IV-4-4
IV-4-3	Response for a Given Memory Function Failure	IV-4-5
IV-5-1	Memory Register Access Privileges	IV-5-4
IV-5-2	CM SECDED Parity Check Matrix	IV-5-6
IV-5-3	Syndrome Codes/Corrected Data Bits	IV-5-7
IV-5-4	Memory Configuration Switch Register	IV-5-8
IV-5-5	Memory Configuration	IV-5-9
V-2-1	Model Dependent CPU Registers	V-2-5
V-2-2	Control Store Addressing	V-2-6
V-2-3	Branch Condition	V-2-8
V-2-4	Maintenance Scan	V-2-11
V-3-1	Multiple Selection	V-3-22
V-3-2	Debug Condition Register	V-3-28

V-4-1	I Mux Select Control	V-4-3
V-4-2	MAP Function Codes for Initialization	V-4-5
V-4-3	MAP Function Codes for Streaming	V-4-10
V-4-4	60-Bit Mode Length Decodes	V-4-13
V-5-1	Ring and Key Changes During Branches	V-5-13
V-5-2	P Register Call Branch	V-5-15
V-5-3	P Register Return - G Key	V-5-16
V-5-4	P Register Return - L Key	V-5-16
V-5-5	MAP Functions	V-5-20
V-5-6	MAP Branch Conditions	V-5-28
V-5-7	MAP Exit Conditions	V-5-29
F-1	Model 810 CP Instruction Timing	F-2
F-2	Model 830 CP Instruction Timing	F-5
C-1	IOU Micrand Fields	C-2
D-1	Microcode Starting Address (19X Hex)	D-10

HARDWARE COMPONENTS AND CONFIGURATION

The system includes the following major components shown in figure I-1:

- One or two central processing units (CPUs)
- Central memory (CM) with a capacity of 2, 4, 8, 12, 16 or 32 megabytes
- Input/output unit (IOU)
 - 10 or 20 peripheral processors (PPs)
 - 4 Integrated Controller Interface (ICI) channels, 7 internal channels and 12 CYBER 170 channels, or
 - 6 Integrated Controller Interface (ICI) channels, 7 internal channels and 10 CYBER 170 channels.

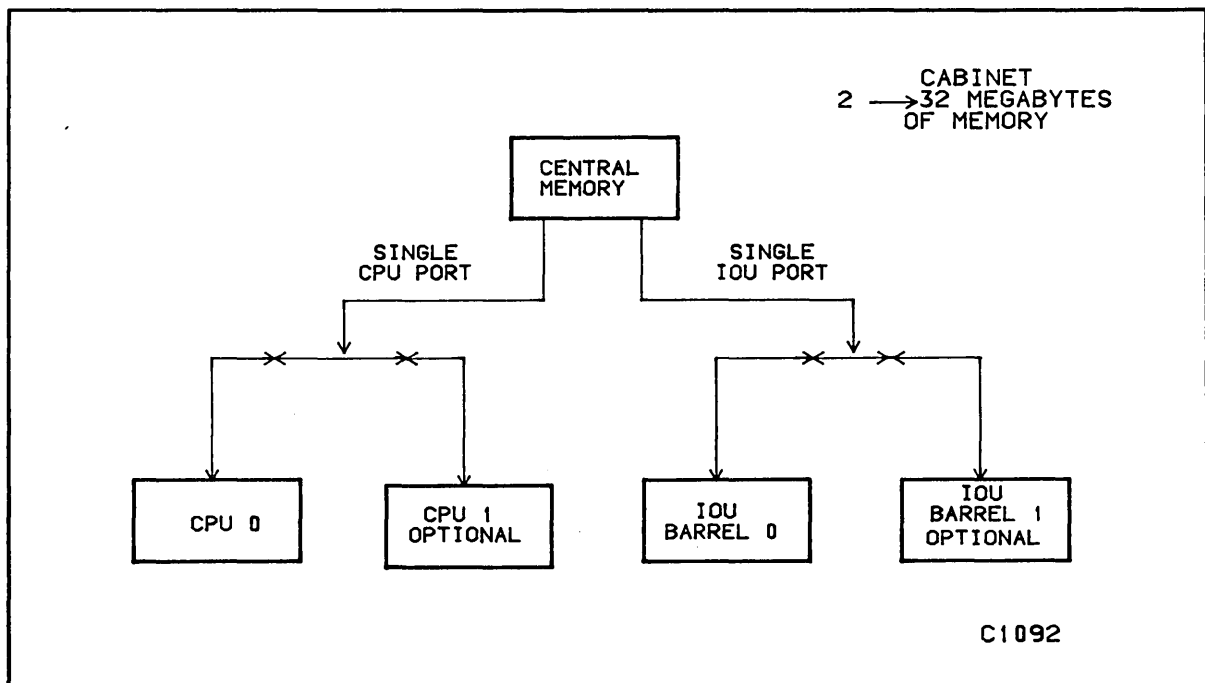


Figure I-1. System Configuration

The following components of the systems are not shown in the diagram:

- System operators console
- Motor-generator set (when used)
- System cabinet
- Power supplies

All mainframe components are air cooled.

CENTRAL PROCESSING UNIT

CPUs 0 and 1 are connected to maintenance access control (MAC) and CM as shown in figure I-2. MAC connects the internal elements of the CPUs through the maintenance channel to the IOU. MAC also provides a CPU data bus and controls to read/write various CPU registers. A single CPU port connects the CPUs to CM. Both CPUs (CPU 1 is optional) can execute the CYBER 170 and CYBER 180 instruction sets.

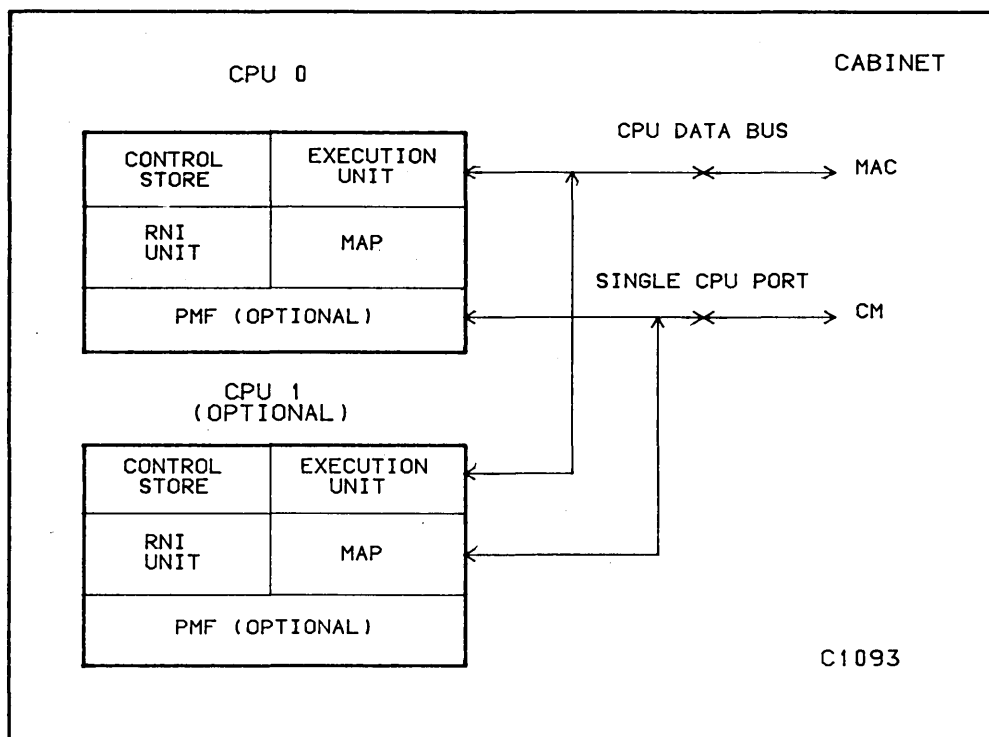


Figure I-2. Central Processor Diagram

The CPU contains the following functional units which perform address translation, arithmetic, logic, and control operations:

- Control store holds the microprogram that controls internal CPU operations. Control store also provides microand addressing facilities.
- Execution units perform logical, arithmetic, shifting, and addressing functions. Random access register files hold process state registers, constant values, and operands for processing.
- Read next instruction (RNI) unit contains instruction buffers and control logic to prefetch sequential instruction words from CM prior to execution. The unit increments the program address register, called P register, according to the decoded instruction length. The execution units also use the RNI unit for stream operations.
- Modified addressing procedure (MAP) unit translates virtual addresses into real memory addresses and performs address validity checking. MAP performs access validation during translation. Microcode routines translate leaving the results in the MAP buffers.
- Performance monitoring facility (PMF) is optional with each CPU. The PMF gives information regarding keypoint class and keypoint code and also provides eight 32-bit counters which can monitor the specified events and states without affecting the performance of the associated processor.

CENTRAL MEMORY

Central memory (CM), shown in figure I-3, has 2, 4, 8, 12, 16, or 32 megabytes (MB) of memory. The 2-MB system is organized into 2 banks; the 4-, 8-, 12-, 16-, and 32-MB systems are organized into 4 banks. Sequential addressing results in a phase bank operation with interleaved mode. Sequential addresses reside in sequential banks.

PURPOSE/FUNCTIONS

CM performs the following functions:

- The two CM ports, CPU and IOU ports, make CM accessible to the CP and each PP.
- Single error correction/double error detection (SECDED) generators generate ECC bits, stored with each word. SECDED checks circuits, corrects single-bit errors, and reports double-bit errors.
- The maintenance channel interface allows a PP access to the CM maintenance registers for system initialization, corrective action, error reporting, and diagnostics.

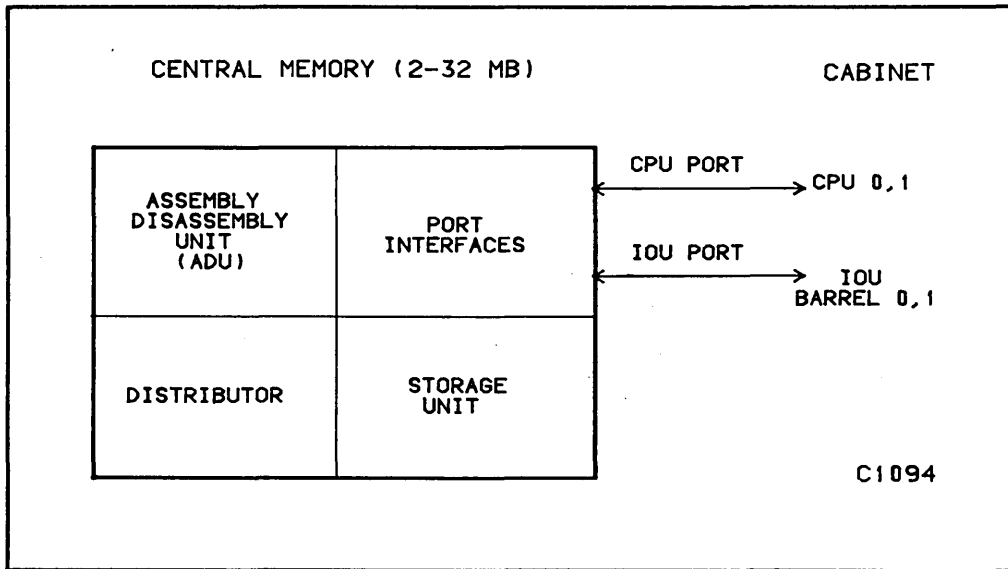


Figure I-3. Central Memory Diagram

CM consists of four major elements: assembly/disassembly unit (ADU), port interfaces, distributor, and storage unit.

ASSEMBLY/DISASSEMBLY UNIT

The assembly/disassembly unit assembles five 12-bit PP words or four 16-bit PP words into a 60- or 64-bit CM word respectively. It disassembles a 60- or 64-bit CM word into five 12-bit or four 16-bit PP words.

PORTS

CM has two ports: IOU port and CPU port. IOU port is connected to IOU; CPU port is connected to CPU. In a dual CPU configuration, both CPUs share one port.

DISTRIBUTOR

The distributor resolves port conflicts and multiplexes data from ports to the storage unit. No port is locked out. The distributor contains the error correction code (ECC) generator, SECCED circuits, and partial write logic. An ECC is generated by the distributor prior to sending the data word to the storage unit; SECCED logic checks data read from the storage unit.

STORAGE UNIT

The storage unit consists of 256K- or 64K-chip memory arrays with data, address, and control logic interfaces. It has two banks for a two-megabyte memory and four banks for other memory sizes. It is a dynamic random access memory with a word length of 72 bits (64 data bits and 8 parity or code bits).

INPUT/OUTPUT UNIT

The CPU or the operator instructs IOU to run PP programs which control all PP operations as shown in figure I-4. The IOU communicates with CM via channels and controls all peripheral communication. The IOU contains the following six major units:

- Deadstart and initialization
- Barrel and slot
- Input/output channels
- IOU maintenance registers
- Two port multiplexer
- CC545 display station controller (optional)

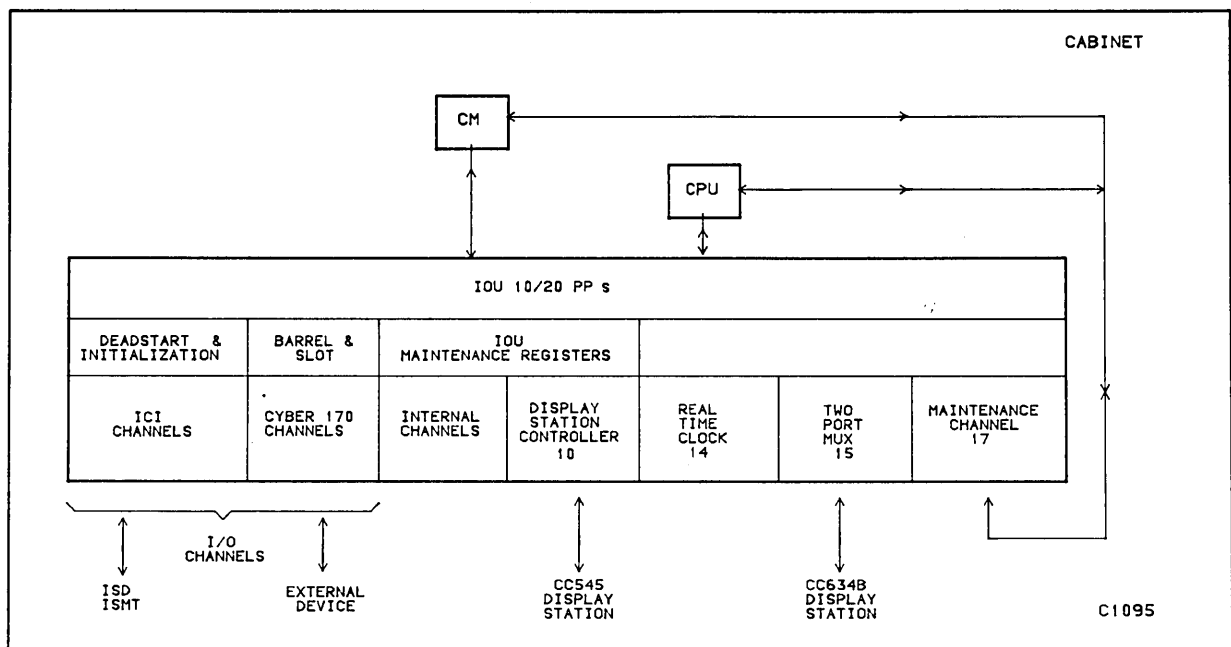


Figure I-4. IOU Diagram

CONFIGURATION

The IOU has 10 or 20 PPs. Each PP is a functionally independent 16-bit computer with 4096 17-bit words (16 data bits plus 1 parity bit) of peripheral processor memory (PPM) and a repertoire of 114 instructions. PPs share access to all channels and one CM IOU port.

SPECIAL CHANNELS

The following are special channels:

- The display station controller (DSC) on channel 10_g, provides a parallel interface for a CC545 display (optional with CC545 console).
- The real time clock is on channel 14_g.
- The two port multiplexer (TPM) on channel 15_g provides the RS232/RS366 serial interfaces needed to connect directly to two terminals.
- The maintenance channel (MCH) on channel 17_g has radial interfaces to other system elements.

STANDARD EQUIPMENT

The following are standard equipment:

- Ten PPs
- First two channel paks which consist of two ICI channels, six CYBER 170 I/O channels, and three internal channels
- Real time clock on channel 14_g
- Two port multiplexer on channel 15_g
- Two maintenance channel radial interfaces on channel 17_g

OPTIONS

The following are options:

- Twenty PPs. (Ten are basic.)
- Three or four channel paks. (Two channel paks are basic.)
- Three or four radial interfaces. (Two are basic.)
- CC545 System Operators Display Station.
- CC545 Display Station Controller which replaces an internal channel.

- Supermini Disk Units (ISD)
- Supermini Tape Units (ISMT)

SECTION II-1

GENERAL DESCRIPTION

=====

The input/output unit (IOU) handles all peripheral communication as well as communication with central memory (CM). An IOU driver services the display station and accepts operator commands. The IOU monitor, which runs in a peripheral processor (PP), communicates with the central processing unit (CPU) by way of CM through a mechanism called the exchange jump. Figure II-1-1 shows the IOU block diagram.

The IOU provides:

- The interface between CM and integrated controller interface (ICI).
- The interface between CM and external devices.
- A parallel interface between the operator and the computer system via an optional CC545 Display Station and display station controller (DSC).
- A serial interface between the operator and the computer system via a CC634B Display Station and the two port multiplexer (TPM).
- Operator accessible maintenance registers and deadstart controls.

CONFIGURATION

IOU is made up of 10 or 20 PPs. Each PP is a functionally independent 16-bit computer with 4096, 17-bit words (16 data bits plus 1 parity bit) of peripheral processor memory (PPM) and a repertoire of 114 instructions. PPs share access to one CM IOU port and bidirectional input/output (I/O) channels.

Each group of 10 PPs is organized in a time multiplexing system called barrel and slot (BAS) which allows PPs to share hardware for arithmetic, logic, and I/O operation without losing speed or independence. A barrel has 10 PPs or ranks (0 through 9). The PPs rotate in order, one rank at a time, around the barrel and through the slot (rank 9). Each rank is a minor cycle time of 50 ns. One trip around the barrel is a major cycle that requires 500 ns.

PPs communicate with CPU using CM data structures and processor exchange instructions. PPs communicate with external devices and other PPs over I/O channels. PPM stores the data and code necessary for each PP's function.

PPM is used as a buffer during data transfer between external devices and CM. This buffering ensures the data transfer is isolated from variations in CM transfer rate.

The assembly/disassembly unit (ADU) reformats I/O words into CM words and CM words into I/O words. Both barrels share one ADU and one IOU port for access to CM. Refer to part IV section two for theory description.

1 21 7
 3 23 11
 4 24 5
 25 27 12
 30 31 13
 33 32

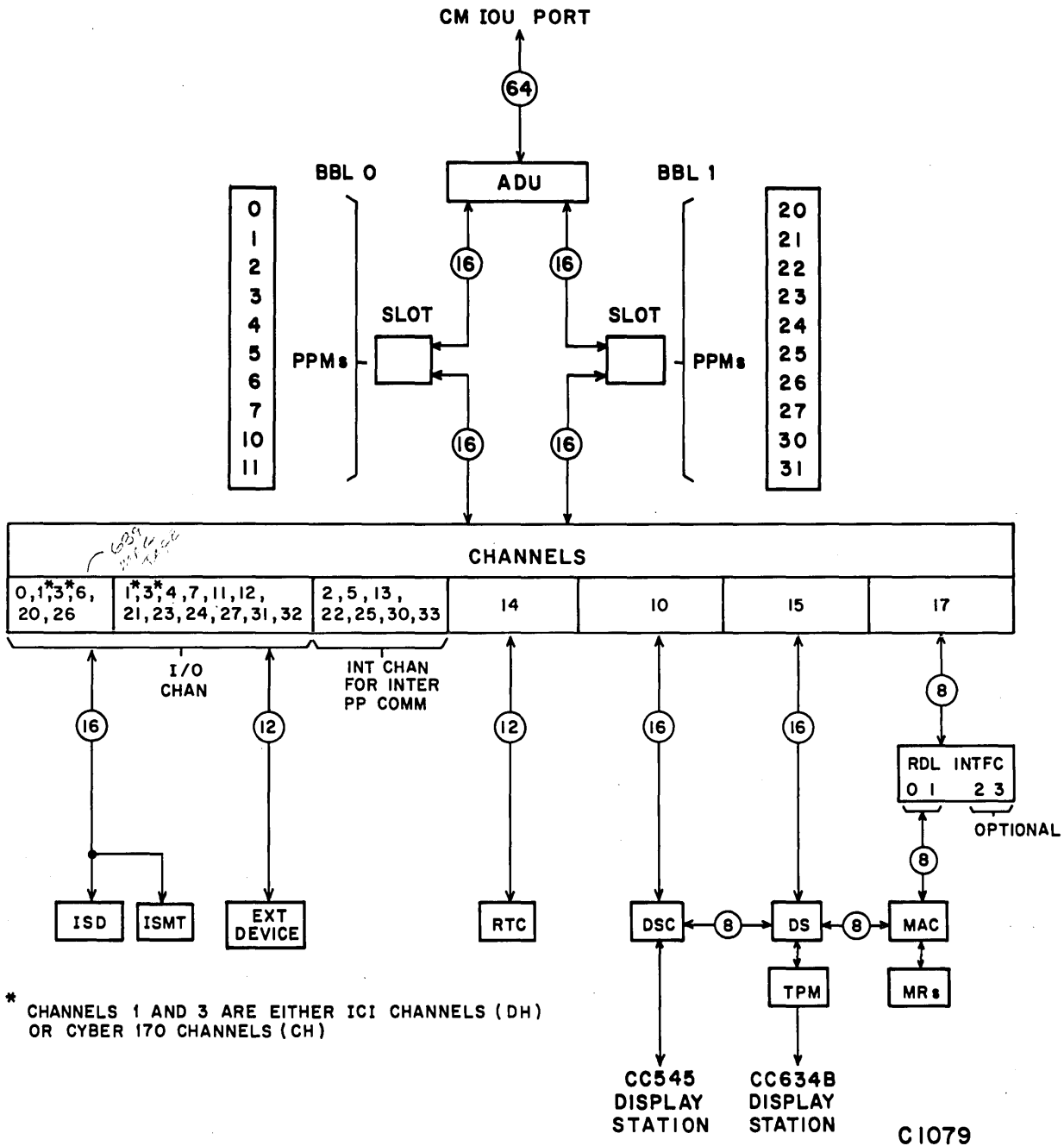


Figure II-1-1. IOU Block Diagram

The IOU contains the following special channels* and interfaces:

- The deadstart control initializes the system.
- The DSC on channel 10 provides a parallel interface for the optional CC545 Display Station.
- The real time clock (RTC) is on channel 14.
- The TPM on channel 15 provides RS232 serial interfaces needed to connect directly with terminals and to provide such features as remote maintenance and remote deadstart.
- The maintenance channel (MCH), channel 17, has radial interfaces to other system elements and includes maintenance access control (MAC) and peripheral environment monitors.

BLOCK DIAGRAM SIGNAL FLOW

Refer to figure II-1-1 for signal flow paths.

The operator communicates with the computer system using the optional CC545 via the DSC on channel 10 or a CC634B via TPM on channel 15.

Maintenance registers are accessible via MAC.

A microprocessor controls the deadstart operation and communication with the TPM. At deadstart time the microprocessor can be used as a deadstart tool.

The following describes the signal flow between a peripheral device and CM.

A 12-bit word from a peripheral device is transmitted over an I/O channel. Leading zeros are added to make up a 16-bit word. During slot time, the specified channel inputs the word to the PP to be stored in the PPM. During the next slot time, the channel inputs another 16-bit word to be stored in the next sequential PPM location. The input operation is complete when all words are stored in successive PPM locations. The words are then sent to the ADU. The ADU assembles 16-bit words into a 64-bit CM word. The PP sends the CM address to CM. The assembled 64-bit word is then stored at the specified CM location. The A register holds the address of the CM location that the assembled 64-bit word is stored in. The remainder of the 16-bit words are assembled and stored in successive CM locations.

The reverse process is followed when reading from CM and outputting to a peripheral device. The 64-bit word is disassembled in the ADU and stored in successive PPM locations. The 12-bit channel words are then output over a channel to the peripheral device.

* All channel numbers are in octal.

MAJOR UNITS

The IOU contains the following six major units:

- Deadstart and initialization
- Barrel and slot
- Input/output channels
- IOU maintenance registers
- Two port multiplexer
- CC545 display station controller

Each of these major units is described in detail in subsequent sections. IOU also contains different macrocells which are used for rank-shifting of registers and other special purposes such as channel conversion and slot execution.

INPUT/OUTPUT CHANNELS

Channels 1 and 3 are CYBER 170 channels in AU130-A; but are ICI channels in the AU129-A option. Input/output channels (0,1,3,4,6,7,11,12,20,21,23,24,26,27,31, and 32) are shown on CHL MLBDs; special channels (10,14,15, and 17) on DSC and SCH MLBDs. Any PP in either barrel may communicate with any other PP over any channel except channel 14. Channels 0,1,3,6,20, and 26 are connected to ISD and ISMT shown in figure II-1-1 and communication is on 16-bit basis. Channels 1,3,4,7,11,12,21,23,24,27,31, and 32 are CYBER 170 channels and communication is on 12-bit basis. Channels 2,5,13,22,25,30, and 33 are internal channels for inter-PP communication only and have no external interfaces.

BARREL AND SLOT

The concept of barrel and slot is shown on BAS MLBDs; the concept of peripheral processor memory is shown on PPM MLBDs. BAS contains the following registers:

- P, the 12-bit program address register
- Q, the 12-bit auxiliary accumulator register
- A, the 18-bit accumulator register
- R, the 22-bit relocation address register
- K, the 7-bit instruction word register
- I, the 9-bit instruction microcode address register
- C, the 12-bit channel data register

Each of these registers is described in detail in section 3.

PPM can store a maximum of 4096 words; each word is 16-bit wide. PPM uses MOS static random access memory (RAM).

CC545 DISPLAY STATION CONTROLLER

Display station controller is shown on DSC MLBDs. It provides communication between PP and CC545 Display Station. The controller is designed to control one CC545 Single-Tube Display Station and to generate displays in either dot or character mode.

TWO PORT MULTIPLEXER

Two port multiplexer is shown on DST MLBDs. TPM is an asynchronous communication interface connected with channel 15. Both ports 0 and 1 have an RS 232-C interface. In addition, port 1 has an RS 366 interface. An eight-position baud rate switch and a four-position port-options switch mounted on the back of the mainframe select baud rate and options for both ports respectively. Only authorized maintenance personnel can access these switches.

TPM supports ASCII code data communication only and has the following features:

- Auto answer
- Remote power control
- Remote deadstart
- Calendar clock

IOU MAINTENANCE REGISTERS

IOU maintenance registers (MRs) are shown on MR MLBDs. The registers indicate fault status in IOU and its environment. The fault status register indicates a failing pak rather than a failing logical unit. MRs also contain bits for installed options and the element identifier. They are accessed through channel 17.

The ten IOU MRs and their addresses are:

- Summary Status (00_{16})
- Element ID (10_{16})
- Options Installed (12_{16})
- Fault Status Mask (18_{16})
- OS Bounds (21_{16})

- Environment Control (30_{16})
- Status (40_{16})
- Fault Status 1 (80_{16})
- Fault Status 2 (81_{16})
- Test Mode ($A0_{16}$)

For further information, refer to section 5 of this part. Use the tab labelled Maintenance Access Hardware.

DEADSTART AND INITIALIZATION

IOU initialization preceeds all other system initialization. Upon completion of initialization, IOU uses the system storage device to provide initialization programs and data for other system elements. A deadstart program is read into PPO to set unique installation parameters. Operator may modify this program. A CC545 or CC634B is required for IOU hardware initialization. By pressing the appropriate button on either console, the microprocessor on CK pak displays the deadstart program. The microprocessor also allows the operator to change the contents of the program, to display contents of PPM, IOU MRs and working registers, and channel status.

The deadstart program has eight different programs stored in ROMs and RAMs to either initialize or test IOU hardware. These programs contain a set of PP instructions and are 16 instruction words long and 16-bits wide. The programs stored in ROMs test IOU hardware. Those programs stored in RAMs initialize IOU hardware and can be modified and stored for future use. The battery located on CK pak provides auxiliary power on these RAMs should system power fail.

SECTION II-2

DEADSTART AND INITIALIZATION

=====

The deadstart procedure initializes the system by selecting a 16-word deadstart program for PP0. When the microprocessor is reset, a stored ROM program runs and displays deadstart information on the CC545 console. The deadstart program also reconfigures PPMs and barrels, and displays error status and maintenance information.

This section lists the deadstart hardware, functions, and selection process and describes the deadstart process in detail. The section also lists long deadstart (LDS) and short deadstart (SDS) hardware and functions, and describes the LDS and SDS processes.

Finally, the features of the DIAGNOSTIC switch on the two port mux box are described.

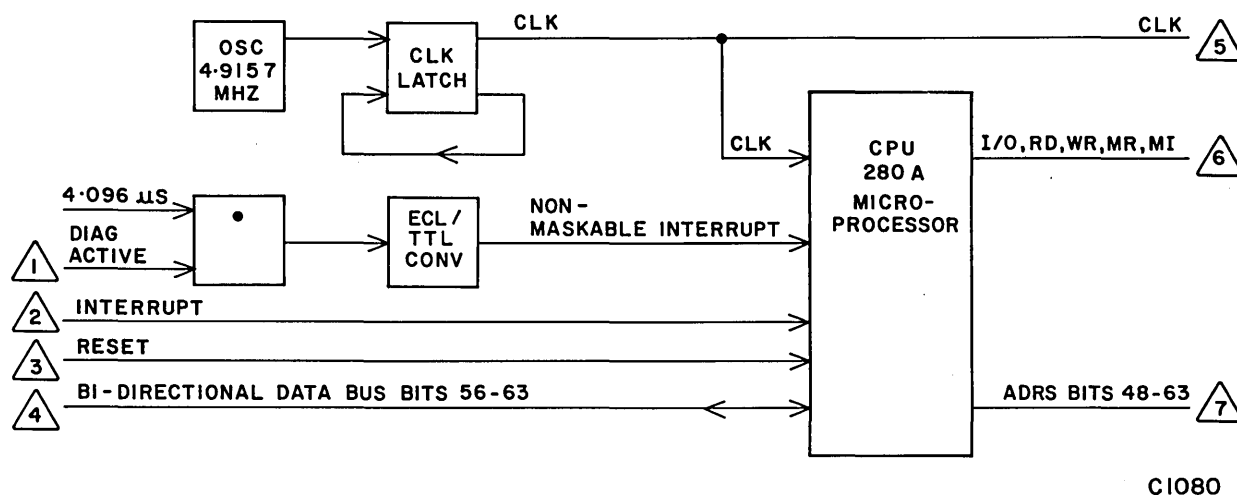
DEADSTART HARDWARE

MICROPROCESSOR

The microprocessor shown in figure II-2-1 (DST 2.0) performs the following:

- Shows the deadstart display and options available
- Displays the A,P,Q, and K registers for up to 20 PPs
- Reconfigures the barrels
- Reconfigures the PPM
- Displays the PPM
- Allows data entry in PPM
- Allows data entry in deadstart display
- Displays maintenance registers
- Allows data entry and maintenance register clearance
- Controls the first-in-first-out (FIFO) buffer memory for the two port multiplexer (TPM)
- Initializes the long deadstart sequence (LDS)
- Initializes the short deadstart sequence (SDS)

- Allows the running of predeadstart diagnostics (chain test)
- Displays channel status registers



1. From two port mux switch
2. From port 0 and port 1 UARTs
3. From deadstart button
4. To and from port 0 and port 1 UARTs, PIO status input ports 0, 1, and 2, microprocessor ROM, microprocessor RAM, and calendar clock
5. To port 0 and port 1 UARTs, PIO status input ports 0, 1, and 2
6. To microprocessor ROM, function decoders 1 and 2, real time clock, port 0 and 1 UARTs, PIO status input ports 0, 1, and 2
7. To microprocessor ROM and RAM, function decoders 1 and 2, calendar clock, ports 0, 1 UARTs, PIO status input ports 0, 1, and 2

Figure II-2-1. Deadstart Microprocessor

The microprocessor is controlled by the following three hardware-controlled signals:

INTERRUPT	Generated by universal asynchronous receiver/transmitter (UART) requiring microprocessor assistance
NON-MASKABLE INTERRUPT	Active when DIAGNOSTIC switch is set to DIAGNOSTIC on two port mux box
RESET	Active when DEADSTART is pressed or during a power recovery

The microprocessor loads data onto and accepts data from the 8-bit microprocessor bidirectional data bus. Fifteen address bits select and control the other microprocessor hardware.

The microprocessor sends five control signals: Memory Cycle, Memory Request, I/O Request, Read, and Write to control the hardware associated with its bus. Refer to section on function decoder.

FUNCTION DECODER 1

Microprocessor address bits 49, 50, and 51 are decoded by the function decoder 1 shown in figure II-2-2 (DST 2.0) to select microprocessor ROM data, microprocessor RAM data, baud rate receiver data, or port select receiver data for input onto the microprocessor bus.

<u>Address Bit</u>			<u>Hex</u>	<u>Device Selected</u>
<u>49</u>	<u>50</u>	<u>51</u>	<u>Address</u>	
0	0	0	0XXX	ROM 0
0	0	1	1XXX	ROM 1
0	1	0	2XXX	ROM 2
0	1	1	3XXX	ROM 3
1	0	0	4XXX	ROM 4
1	0	1	5XXX	RAM*
1	1	0	6XXX	baud select**
1	1	1	7XXX	port select**

* ANDed with Memory Request signal

** ANDed with Memory Request and Read

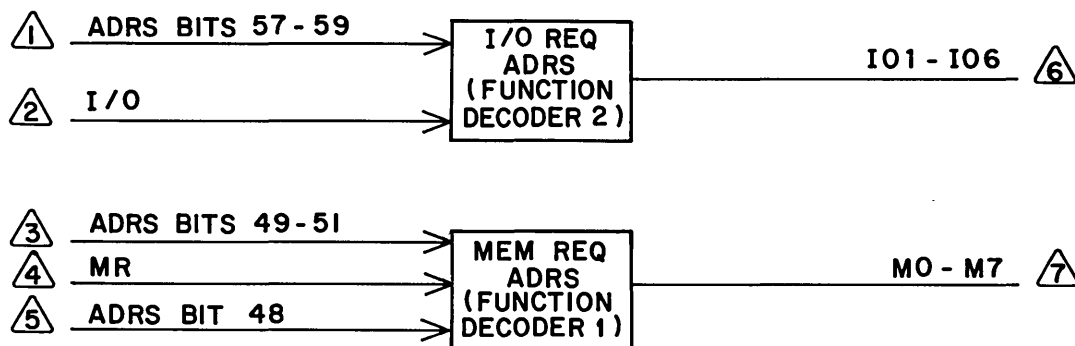
X Don't care

FUNCTION DECODER 2

Microprocessor address bits 57, 58, and 59 (bit 56 included to complete the hex address) are decoded by the function decoder 2 shown in figure II-2-2 (DST 2.0) to select one of the following: parallel input/output port (PIO) 0, 1, and 2, universal asynchronous receiver/transmitter (UART) 0 and 1 and the calendar clock. The function decoder 2 outputs if the I/O Request signal sets.

<u>Address Bit</u>				<u>Hex</u>	<u>Device Selected</u>
<u>56</u>	<u>57</u>	<u>58</u>	<u>59</u>	<u>Address</u>	
X	0	0	1	XX1X	PIO 0
X	0	1	0	XX2X	PIO 1
X	0	1	1	XX3X	PIO 2
X	1	0	0	XX4X	UART port 0
X	1	0	1	XX5X	UART port 1
X	1	1	0	XX6X	calendar clock

X Don't care



C1081

- 1-5. From deadstart microprocessor
6. To calendar clock, port 0 and 1 UARTs, PIO status input ports 0, 1, and 2, microprocessor RAM
7. To microprocessor ROM and RAM, PIO ports 0, 1, and 2

Figure II-2-2. Deadstart Function Decoders

MICROPROCESSOR ROM

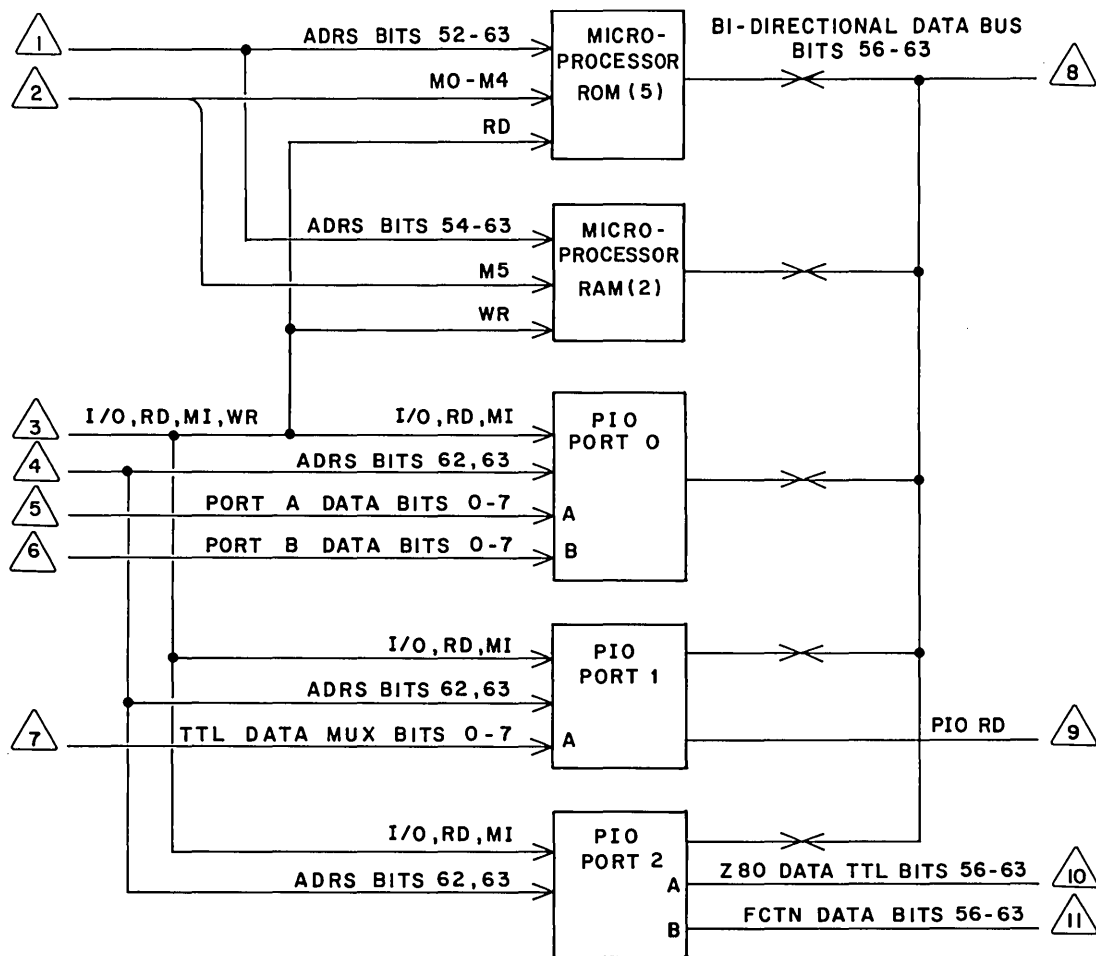
The 5 ROMs in figure II-2-3 (DST 2.0) contain the microprocessor software and deadstart programs and LDS program. ROMs are:

- Selected by function decoder 1
- Read enabled by Memory Request and Read
- Addressed by bits 52 through 63

MICROPROCESSOR RAM

The two RAMs in figure II-2-3 (DST 2.0) store microprocessor data and customized programs. RAMs are:

- Selected by function decoder 1 and Memory Request
- Write enabled by Write and +5V-OK
- Addressed by bits 54 through 63



C1082

- 1 & 4. From deadstart microprocessor
2. From function decoder 1
3. From deadstart microprocessor
- 5 & 6. Status data
7. Data from FIFO port 0 and 1, PPM, or keyboard
8. Bidirectional data to or/from deadstart microprocessor, UART's, and calendar clock
9. To FIFO control
10. Output data to system
11. Function data to system

Figure II-2-3. Deadstart ROM, RAM, and PIO Ports 0, 1, and 2

PARALLEL INPUT/OUTPUT PORT

The functions of the three parallel input/output ports (PIO) in figure II-2-3 (DST 2.1) are:

- PIO 0 ports A and B collect status information during operation
- PIO 1 port A inputs data from FIFO port 0 or port 1, PPM, or keyboard
- PIO 2 ports A and B output data or function codes to the system

The PIOs are:

- Selected by function decoder 2
- Read/write enabled by address bits 62, 63 as follows:

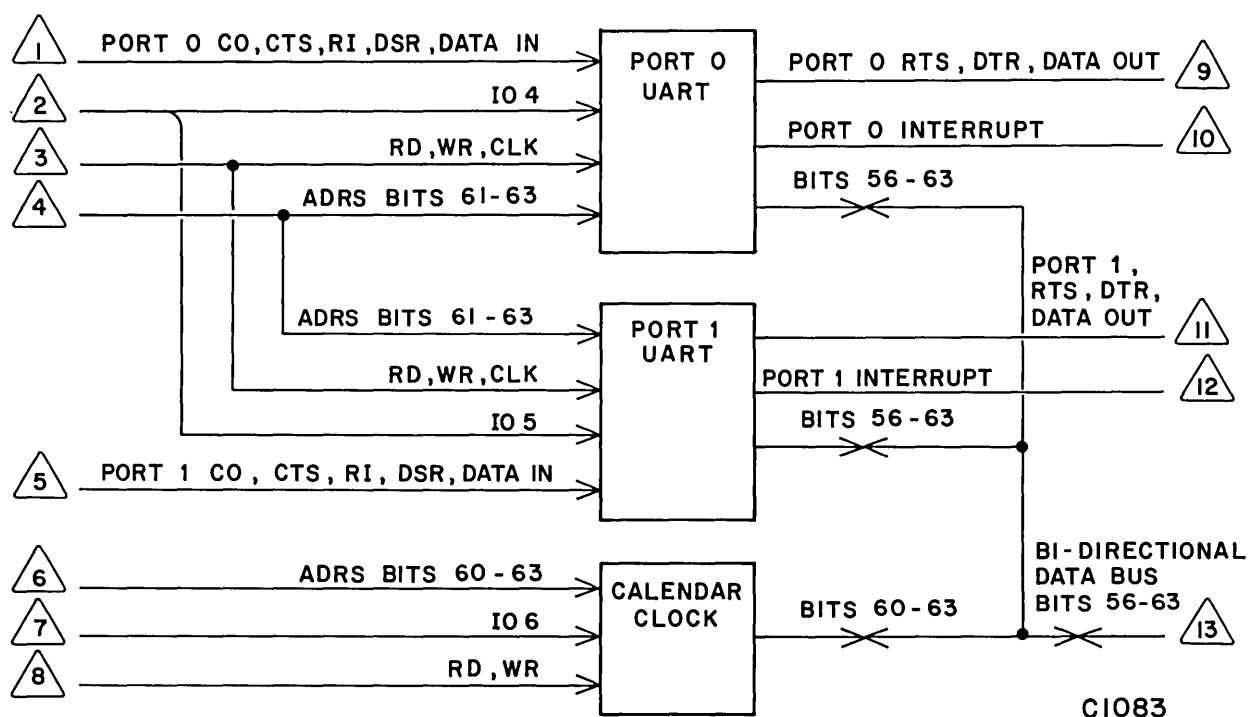
<u>Address Bit</u>	<u>Function</u>
62 set	Control
62 not set	Data select
63 set	Port B
63 not set	Port A

- Function selected by Read, Memory Cycle, and I/O Request as follows:

	<u>SIGNAL</u>		<u>FUNCTION</u>
<u>Read</u>	<u>Memory Cycle</u>	<u>I/O Request</u>	
Active	Active	Active Active Active	To microprocessor to PIO interrupt

UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER

The universal asynchronous receiver/transmitter (UART) shown in figure II-2-4, (DST 2.1) assembles and disassembles data between the TPM terminal and the microprocessor. When the UART is outputting to a terminal, parallel data (8 bits) from the microprocessor are serialized and transmitted to the terminal. When the UART is inputting from the terminal, serial data from the terminal is input to the UART and parallel data is sent to the microprocessor.



- 1 & 5. Data from terminal
- 2 & 7. From function decoder 2
- 3 & 8. From deadstart microprocessor
- 4 & 6. From deadstart microprocessor
- 9 & 11. Data to terminal
- 10 & 12. To deadstart microprocessor
- 13. To deadstart microprocessor, ROM, RAM, and PIO ports 0, 1, and 2

Figure II-2-4. Universal Asynchronous Receiver/Transmitters

The two UARTs are:

- Selected by function decoder 2
- Function selected by address bits 61, 62, and 63 (bit 60 included to complete the hex address)
- Read enabled by READ
- Write enabled by WRITE

<u>DLAB</u>	<u>Address Bit</u>				<u>Hex Address</u>	<u>Register Selected</u>
	<u>60</u>	<u>61</u>	<u>62</u>	<u>63</u>		
	<u>A2</u>	<u>A1</u>	<u>A0</u>			
0	X	0	0	0	XXX0	Receiver buffer (read), transmitter holding register (write)
1	X	0	0	0	XXX0	Divisor latch (LSB)
0	X	0	0	1	XXX1	Interrupt enable
1	X	0	0	1	XXX1	Divisor latch (MSB)
X	X	0	1	0	XXX2	Interrupt identification (read only)
	X	0	1	1	XXX3	Line control
	X	1	0	0	XXX4	MODEM control
	X	1	0	1	XXX5	Line status
	X	1	1	0	XXX6	MODEM status
	X	1	1	1	XXX7	None

X Don't care

CALENDAR CLOCK

The calendar clock (DST 2.0) operates on its own 32.76-kHz crystal. For more information refer to section II-6. The calendar clock is:

- Selected by function decoder 2
- Write enabled by Write and +5V OK
- Read enabled by Read
- Function selected by address bits 60 through 63

CC545 CONSOLE

Pressing DEADSTART on the CC545 causes the deadstart display to appear on the CC545 console. No Master Clear is sent to the PPs at this time.

CC555 TERMINAL

CNTL G causes an alert message to appear on the terminal. CNTL R causes the deadstart display to appear on the CC555 terminal. No Master Clear is sent to the PPs at this time. However, if there is a PP communicating with the TPM at this time that PP may hang. It will be released when an LDS or an SDS is performed.

For deadstart commands entered from the terminal keyboard refer to the Hardware Operator's Guide and the Maintenance and Parts Hardware Maintenance Manual.

POWER-ON MASTER CLEAR

The POWER-ON MASTER CLEAR performs the following:

- Sets R register to zero for all PPs
- Issues a 100 millisecond Master Clear to all channels and PPS (see master clear description below)
- Resets microprocessor

MASTER CLEAR

When a Master Clear is issued, the channels respond as follows:

Channel 0 - inactive, empty, channel flag clear, and channel error flag clear.

Channel 14 - fixed status: active, full, channel flag set, and channel error flag set (MASTER CLEAR has no effect).

Channels 15 and 17 - active, empty, channel flag clear, and channel error flag clear.

User channels - active, empty, channel flag clear, and channel error flag clear.

Optional channels not installed - inactive, empty, channel flag clear, and channel error flag clear.

Master Clear is transmitted on all installed user channels (00 to 13; 20 to 33).

Master Clear initializes the PP registers as follows:

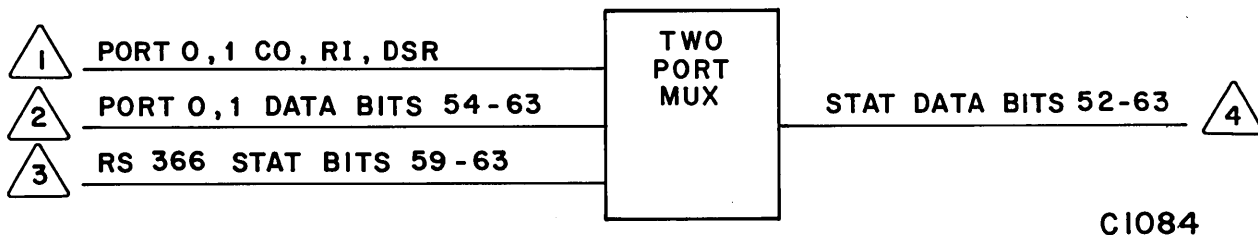
<u>Register</u>	<u>Value</u>	<u>Comments</u>
K	0071	Op code for a block channel input instruction.
I	776	Microcode address of first micrand.
P	7777	
A	010 000	
Q	0,1,...,11	Each Q register is loaded with a number corresponding to its PP number: PP 00 = 00, PP 01 = 01, etc.
C	N/A	Not initialized.
R	N/A	Initialized to zero only by Power-On Master Clear or by the master clear ADU MAC function.

TWO PORT MULTIPLEXER

The TPM shown in figure II-2-5 (DST 2.2) is a unit in the mainframe which provides two RS-232-C interfaces, called port 0 and port 1. Display terminals which have RS-232-C interfaces may be connected with one or both of these ports, either directly or through compatible modems. The ports contain logic to communicate with terminals or modems. For details refer to section II-6.

Normally, the TPM provides an interface between the operating system software and a low-speed terminal or maintenance terminal. The display on the screen of the terminal and interpretation of commands entered into the terminal are under control of the software running in the PPs. The software interfaces to the TPM by means of channel 15. However, the deadstart microcomputer may be activated to take over either port and provide the deadstart display. If the special character CNTL G is detected, it activates the microcomputer.

The deadstart display may be activated by a terminal on either port.



1. From terminal
2. From PIO port 2
3. From terminal
4. To channel 15 and deadstart

Figure II-2-5. Two Port Mux

NOTE

When data is entered using EP, ER, or editing commands, the screen image is not updated immediately. The display is not refreshed during data entry but is resumed upon carriage return. When using low baud rates, the refresh interval is significant.

If the deadstart display is activated for that port, the message OPERATOR ACCESS ENABLED is displayed on the terminal. After the last character of this message is sent to the terminal, the microcomputer inspects the keyboard for a CNTL R character. The combination of CNTL G followed by CNTL R is similar to pressing DEADSTART on the CC545 console.

It is possible to use the deadstart display on either port to inspect internal data on a running system with the following restrictions:

- The software currently running in the PPs must not attempt to use channel 15 (TPM or wall time clock features).
- The following displays and commands are potentially destructive to the IOU:
 - PPM Display (PM)
 - Maintenance Register Display (MR)
 - Enter PP Memory Command (EP)
 - Enter Maintenance Register Command (ER)
 - Any of the commands to reconfigure, idle, deadstart, or power off

DEADSTART HARDWIRED

Pressing DEADSTART on the console resets the following:

- Microprocessor (DST 2.0)
- RS 366 Register (DST 2.3)
- Port 0 and Port 1 UARTs (DST 2.1)
- RS 366 Interface Enable FF (DST 2.3)
- Second Parcel FF (DST 2.4)
- Select CC545 FF (DST 2.5)
- Medium Character FF (DST 2.5)
- Hold Function FF (DST 2.8)
- Read PIO FF (DST 2.7)
- Keyboard Full FF (DSC 2.0)

DEADSTART SOFTWARE

When DEADSTART is pressed, the microprocessor (DST 2.0) is reset. All the internal registers are set to zero and the address lines and control lines are pulled low.

After the microprocessor is reset, it automatically addresses location zero and sends the control signals Memory Request and Read. Microprocessor ROM 0 is selected and location zero is addressed. The contents of ROM location zero are put on the 8-bit bidirectional microprocessor bus and read by the microprocessor. The microprocessor then executes this command and automatically increments its program counter to read and execute the next command.

The initial deadstart software performs the following functions:

- Reads and executes microprocessor ROM 0 (DST 2.0) address zero.
- Determines cause of reset (UART port 0, UART port 1, or CC545 console).
- Initializes PIO 0 and 1 to input mode and PIO 2 to output mode (DST 2.1).
- Initializes UARTs to 7-bits data, 2-bits stop, and even parity.
- Reads BAUD RATE SELECTION switches and sets rate for each TPM port.
- Resets software status bytes and flags.
- Checks for microprocessor RAM battery failure.
- Initializes CC545 console.
- Outputs initial deadstart display on console.
- Waits for keyboard input.

LONG DEADSTART SEQUENCE

The long deadstart sequence (LDS) tests peripheral processors 0 to 4 (PP0 to PP4) using a program stored in the LDS ROM (DST 2.0).

The LDS is initiated by typing the character L when the console is displaying the initial deadstart display. The microprocessor program recognizes the character L as a special character and the program performs a long deadstart task.

The following operations are performed when LDS is initiated:

- When the character L is typed on the console keyboard the character is received by the keyboard data receiver (DSC 2.0), clocked through the keyboard data receiver and the keyboard data register, and converted into ASCII in the display code to ASCII ROM (DSC 2.0). The ASCII data is selected from the interface select mux (DST 2.7), converted to TTL levels, and input to PIO 1 port A (DST 2.1). The microprocessor reads PIO 1 port A and activates the LDS task.
- During the LDS task the microprocessor (DST 2.0) outputs function code 2 through the PIO 2 port B to the microprocessor function register (DST 2.5). The function is decoded in the function decoder (DST 2.5) to initialize the hardware. The LDS program from ROM is transferred to PPO memory and then verified. The starting address stored from RAM is sent to PP and GO LDS function is sent. The Deadstart Sequence in Progress (DST 2.9) signal starts the LDS branch test timer circuit. The signal also sets the LDS Reset Timing FF. The FF output is controlled through the LDS control circuit which sets the new function register in the I register macrocell (BAS 2.7).
- The microprocessor (DST 2.0) outputs the first LDS ROM address (6000) and addresses the LDS ROM (DST 2.0).
- The LDS ROM (DST 2.0) data input to the PP. The PP performs this instruction, then reads and performs the next LDS ROM instruction.
- The first part of LDS is timed to location 6200 after 41 slot times. Location 6200 is selected and a comparator compares it to the P register address. When location 6200 is reached the LDS control circuit (DST 2.9) checks the LDS branch test timer circuit to determine correct PP operation.
- The second part of LDS is complete when location 7776 is reached. The LDS control circuit (DST 2.9) generates the LDS End signal.
- At the end of LDS a short deadstart sequence is initiated (see below).

SHORT DEADSTART SEQUENCE

The short deadstart sequence (SDS) initializes all the PPs and runs the program displayed on the console.

SDS is initiated by typing the character S when the console is displaying the initial deadstart display or automatically upon completion of LDS.

The following operations are performed when SDS is initiated:

- The A register in each PP is set to 10000g by the A register channel data mux (CA 3.0).
- The P register in each PP is set to 7777g by the PQ mux (CSM 3.1).

- The Q register in each PP is set to the PP number.
- The K register in each PP is set to 071g via K barrel select mux (BAS 2.6).
- The program on the deadstart display is loaded into PPO locations 1 to 20. The microprocessor reads the deadstart display programs and sends the program via the PIO 2 port A to PPO.
- Control is transferred from the microprocessor to the PP which runs the program shown on the deadstart display.
- The contents of the deadstart display determines the task that PPO will perform (start the loading of the operating system).
- At completion of deadstart the microprocessor is made available to the TPM. After the operating system is loaded the microprocessor services the TPM.

DIAGNOSTIC SWITCH

The diagnostic switch on the two port mux box displays characters for test purposes diagonally on the CC545 console or the CC555 terminal. When the switch is set to DIAGNOSTIC, it is ANDed with a 4-microsecond pulse from the real time clock producing the Non-maskable Interrupt signal (DST 2.0). When the microprocessor receives this signal it starts the non-maskable interrupt routine located at microprocessor ROM location 66g.

The CC555 or Viking terminal on TPM ports 0 and 1 or the CC545 console are selected by the PORT OPTIONS switches. If either port is active the diagnostic routine runs on that port, if both ports are active the diagnostic routine runs on port 0. During the diagnostic routine the microprocessor performs as follows (CC545 console selected):

- PIO 0 ports A and B are set to input.
- PIO 1 ports A and B are set to input.
- PIO 2 ports A and B are set to output.
- UART ports 0 and 1 are initialized to 7 data bits, 2 stop bits, and even parity.
- A test pulse is output on UART port 0 pin 34, 1CK0, location J12, at test point 08.
- Microprocessor reads PORT OPTIONS switches via the function decoder 1 and the port select receiver to determine if TPM port 0, port 1 or the CC545 console is enabled to run the diagnostics routine (in this example TPM port 0 and port 1 are disabled).
- A character is read from keyboard buffer register.

- The character is displayed diagonally across the console.
- The program then halts and waits for the next Non-maskable Interrupt signal which occurs every 4 microseconds.

SECTION II-3

BARREL AND SLOT

=====

The barrel and slot (BAS) consists of seven registers (A, R, Q, P, K, I, and C), each of which has ten ranks from 0 through 9. Information in these registers moves from one rank to the next at a uniform 20-megahertz (50-ns) rate, providing a multiplexed system of 10 PPs.

The data shifts through the 10 ranks, advancing one rank (minor cycle) at a time, until it reaches the slot. The slot, which is rank 9, is the execution hardware. When data leaves the slot, it enters rank 0 register and begins another 500-ns major cycle trip around the barrel.

When data enters the slot, a portion of the instruction for that data is executed. The slot performs arithmetic and logic operations and program address manipulation. Complete execution of an instruction may require a series of trips around the barrel.

CT macrocells are used to rank-shift registers. A barrel counter or chip address generator that increments at 50-ns intervals controls each shift.

Each PP has 4096 words (one word is 16 data bits plus one parity bit) of memory. Each PPM pak (CM) has 17 data bits and 12 address bits for each group of 10 PPs. PPM uses transistor-transistor logic (TTL).

A BARREL/REGISTER

Size: 18 data bits plus 1 parity bit

	46	63
Bit Numbering	_____	

The A register (BAS 2.0 and 2.1) holds one operand for arithmetic, logic, or selected I/O operations. Various instructions operate on 6, 12, 16, and 18 bits of A. The 18 A data bits may be:

- An arithmetic or logical operand
- A CM address or part of a CM address depends on bit 46
- An I/O function or data word
- A word count for block I/O instructions

The A shifter macrocell (CA 3.0) performs a left circular shift or a right end-off shift. The A adder is an 18-bit ones-complement adder. Parity is checked at both inputs to A adder: at input A (A barrel or channel data) and at input B (PPM).

Parity in A barrel is regenerated after the register has been loaded, or after any arithmetic or logical manipulation of the register data. If data in the A register is unchanged, parity is carried through. On the shift instruction, parity is only regenerated after a right end-off shift. The A parity mux (BAS 2.0) selects the A register parity.

A ADDER

The A input to the A adder (BAS 2.1) is A register Rank 9 bits 46 through 63.

A adder B operand mux (BAS 2.1) selects the B input to the A adder. Inputs to this mux are Q Register Rank 9 bits 58 through 63 and PPM Data Rank 9 bits 48 through 63.

The A adder code bits 0 through 4 are micrand bits 8 through 12. Table II-3-1 shows the A adder code.

TABLE II-3-1. A ADDER CODE

A Adder Code Bits 0-4	A Adder Code
0	F=A
4	F=A AND B
5	F=B
10	F=A AND (NOT B)
11	F=A XOR B
12	F=NOT B
26	F=A minus B
31	F=A plus B

Data from the A adder clocks through the A rank 0 register (BAS 2.1). The A to A mux (BAS 2.0) selects either A Bypass bits 46 through 63 or A Register Rank 0 bits 46 through 63 from A adder. The output of the A to A mux enters the A shifter macrocell.

A SHIFTER MACROCELL

The A shifter macrocell (CA 3.0) has an 18-bit port for the A register data and a 16-bit port for the channel data from the channel input conversion macrocell (BAS 2.10). The macrocell performs three functions:

- Selects data from either port with the A register channel data mux (CA 3.0)
- Shifts A register data left or right according to Shift bits 0 through 5

- Forces A register deadstart value of 010000g on the output port

The shift count, PPM Data Rank 9 bits 58 through 63, addresses the shift ROM (BAS 2.1). The ROM output, Shift bits 0 through 5 clocks through the shift rank 0 register (BAS 2.1) and controls the shifter macrocell.

The data flows through three muxes (CA 3.0). The rank A shift mux 1 performs a right shift of 0, 1, 2 or 3 positions specified by Deadstart or Shift bits 3 and 4. Rank A shift mux 2 is a five-way mux which performs a right shift of 0, 4, 8, 12 or 16 positions on the mux 1 output specified by Deadstart or Shift bits 0 through 2. The A register channel data mux selects either the channel data or the shifted output data from mux 2 specified by Channel Data to A Rank 0.

A low on Deadstart Rank 0 forces the A register channel data mux to select mux 2 data. The shift bits, doing a right shift of greater than 18, force all zeros at the output of mux 2. Only Shifted A2 bit 51 is forced to a one resulting in the A register deadstart value of 010000g.

Zero Count bit 0 indicates a right shift and forces gates 1 through 4 to zero. The shifter always performs right shifts and so a left shift must be converted to its equivalent right shift in the shift ROM. Zero Count bit 0 must be inactive. The output from the shifter macrocell enters the A rank 1 through 8 register macrocell (BAS 2.0).

A RANK 1-8 REGISTER CHIP 1.2 MACROCELL

The A Rank 1 through 8 Register Chip 1.2 Macrocell is a 20-bit register file chip (CT 3.0) with independent write and read ports. The write port consists of two address bits, 20 data bits, a write enable, and a clock. The read port consists of two address bits, 20 data bits, and an output enable.

The barrel counter (BAS 2.0) counts from 0 through 7 at 50-ns intervals. Barrel Counter bit 0 is Write/Output Enable (CT 3.0). Barrel Counter bits 1 and 2 are Read/Write Address bits 0 and 1. Write Address bits 0 and 1 are decoded and gated with T1 to form Write Words 0 through 3 which are used to enable word 0 through 3 registers. Each of these registers inputs the data out mux. The Read Address bits 0 and 1 select one of the four inputs.

Word 0 through 3 registers are quad latches organized as four 20-bit wide words. Data is written into the register (specified by the write address) when the Write Enable is high and T1 is low.

To read the CT macrocell, Read Address bits 0 and 1 access the desired word and the Output Enable is low. If the Output Enable is high, all Data Out bits are forced low allowing the outputs to be emitter-ANDed with another CT chip.

R BARREL/REGISTER

Size: 22 data bits plus 2 parity bits

Bit Numbering:

36	57
----	----

The R register (BAS 2.2) holds the relocation address for all CM read/write and exchange instructions. The R+A adder is a 22-bit adder. It adds A Select Rank 9 bits 47 through 57 to R Select Rank 9 bits 36 through 57 (BAS 2.3). It is a two's-complement adder with no overflow indication.

The two parity bits are: one bit for the least significant 12 bits (46 to 57), one bit for the most significant 10 bits (36 through 45).

New data is loaded into the R register from two different sources: PPM Data Rank 9 bits 52 through 63, and all zeros. The PPM data is from the PPM Rank 9 register (BAS 2.2). Zeros are selected to power-on master clear the R register.

R+A ADDER

The R+A adder (BAS 2.3) adds A Select Rank 9 bits 47 through 57 to R Select Rank 9 bits 36 through 57. The result from the adder is the CM relocation address. Refer to section IV-2, CM Addressing.

R MUX

The R mux (BAS 2.2) loads PPM rank 9 data into the R register according to Load R Upper, Load R Lower Rank 9, and ADU/Power On MC Rank 9. Load R Upper and Load R Lower are Micrand Rank 8 bits 0 and 1 respectively. Four conditions are possible:

- R Register Rank 9 bits 36 through 57 pass through the mux if no control is active.
- PPM Data Rank 9 bits 52 through 63 are loaded into the lower order 12 bits of R register if Load R Lower Rank 9 is active.
- PPM Data Rank 9 bits 54 through 63 are loaded into the higher order 10 bits of R register if Load R Upper Rank 9 is active.
- Zeros are loaded if ADU/Power On MC is active.

RA MUX

The RA mux (BAS 2.3) selects upper or lower data from either the R+A adder or the R select rank 0 register for input to the Y mux (PPM 2.2). The RA mux select bits 0 and 1 are EC Register bit 61 and R Upper Select from branch 2 ROM (BAS 2.7).

R RANK 0-7 REGISTER CHIPS 1-3 MACROCELL

Three chips are needed for the R register (CT 3.1). Chips 1 and 2 are for R data bits 36 through 55; chip 3 for R data bits 56 and 57. Refer to A Rank 1 through 8 Register chip 1, 2 Macrocell for description.

Q BARREL/REGISTER

Size: 12 data bits plus 1 parity bit

Bit Numbering

52	63

The Q register (BAS 2.4) holds data for several functions:

- Address of an operand during direct, indirect, and memory address mode instructions (30s, 40s, and 50s)
- Word count (number of CM words) for block CM read/write instructions
- Channel number during all I/O instructions
- Address in all jump instructions

If the most significant bit (58) of the field is set, sign extension is performed on that field during relative jump instructions. This effectively subtracts the value of d from Q register.

Parity is checked at both inputs to Q adder: A input (Q barrel or P barrel) and B input (PPM). Parity in Q barrel is regenerated after the register is loaded or after any manipulation of the register data. If data in Q register is unchanged, parity is carried through.

New data is loaded into the Q barrel from three different sources: P register, PPM data, and deadstart channel number.

Q SLOT RANK 0 REGISTER CHIP 1-3 MACROCELL

Three chips are needed for Q slot (CSM 3.0). Each chip contains a 4-bit slice of Q register. Chip 1 contains bits 52 through 55; chip 2 contains bits 56 through 59, and chip 3 contains bits 60 through 63.

PQ Mux

The PQ mux (CSM 3.0) passes through Q barrel data or selects data from P register rank 9. P to Q Control (Micrand bit 2) selects the P register data.

Q Adder

The Q adder (CSM 3.0) is a 12-bit ones-complement adder controlled by Q Subtract Rank 9. The A input to the Q adder is bits 52 through 63 from the PQ mux. The B input is a combination of the f and d fields (PPM Data Rank 9 bits 52 through 63).

ADQ-B Mux 1 and 2

The ADQ-B mux 1 and 2 provide the 4-bit B input to the Q adder (CSM 3.0). Chip Number Select bits 0 and 1 select one of the three macrocell chips. B Operand Control bits 0, 1 and P to Q Control control the two muxes. Table II-3-2 shows the selection.

TABLE II-3-2. ADQ-B MUX 1 AND 2 SELECTION

B Operand Control Bit			ADQ-B Bits 52-63
O (F To Q)	1 (D To Q)	P TO Q	
0	0	0	0001g
0	1	0	0,0,0,0,0,0, PPM Data Rank 9 bits 58-63
0	1	1	PPM Data Rank 9 bit 58 (Extended), PPM Data Rank 9 bits 58-63
1	1	0	PPM Data Rank 9 bits 52-63

ADQ Mux

The ADQ mux (CSM 3.0) selects either ADQ B bits 52 through 63 or ADQ-Sum bits 52 through 63. Load Q Rank 9 (Micrand bit 14) controls the mux.

Q Mux 1 and 2

If active, Q to Q Rank 0 to Q mux 1 (CSM 3.0) selects Q Bypass Rank 0 bits 52 through 55.

If active, Deadstart Rank 0 to Q mux 2 selects Deadstart Channel Number Barrel 0 bits 59 through 63.

Q RANK 1-8 REGISTER CHIP 1-3 MACROCELL

Three chips are needed for the Q Register. Chip 1 is for Q data bits 52 through 57. Chips 2 and 3 are for Q data bits 58 through 63. Refer to A Rank 1-8 Register Chip 1, 2 Macrocell for description.

P BARREL/REGISTER

Size: 12 data bits plus 1 parity bit

Bit Numbering:

52	63

The P register (BAS 2.5) contains the program address. The P incrementer (CSM 3.1) increments or decrements by one in twos-complement logic.

The P register does not always contain the program address; it holds and increments the PPM address during:

- Block I/O instructions
- Block CM read/write instructions

Parity is checked at the input to the P incrementer. Parity is regenerated after the register is loaded, or after any manipulation of the register data. If the data is unchanged, parity is carried through.

P SLOT RANK 0 REGISTER CHIP 1-3 MACROCELL

The P register shares the same three slot macrocell chips with the Q register (CSM 3.1). Chip 1 contains P data bits 52 through 55; chip 2 contains P data bits 56 through 59; chip 3 contains P data bits 60 through 63.

ADP-A Mux

FD to P to the ADP-A mux (CSM 3.1) selects either PPM rank 9 or P register rank 9 data. The output of the mux is the A input of the P incrementer. The P incrementer increments or decrements according to Chip bit 0 or Decrement P Rank 9. The outputs of ADP-A mux and P incrementer clock through P register rank 0 register.

P Mux Circuit

Deadstart Rank 0 and P to P Rank 0 are select bits 0 and 1 to the P mux circuit. Table II-3-3 shows the selection.

TABLE II-3-3. P MUX CIRCUIT SELECTION

Deadstart Rank 0	P To P Rank 0	P Mux Deadstart Data Bits 52-55
0	0	P Adder bits 52-55
0	1	P Bypass bits 52-55
1	0	MAC Data bits 52-55
1	1	MAC Data bits 52-55

PQ Mux

P Mux LSB and Deadstart Rank 0 control the selection of the PQ mux (CSM 3.1). Table II-3-4 shows the selection.

TABLE II-3-4. PQ MUX SELECTION

Deadstart Rank 0	P Mux LSB	P Data Bits 52-63
0	0	P Bypass bits 52-63
0	1	Q Mux 1 bits 52-63
1	0	7777 ₈
1	1	MAC Data bits 52-63

G and GP Mux

The G and GP mux (CSM 3.1) select G data (PPM address) to PPM. Table II-3-5 shows the selection.

TABLE II-3-5. G DATA SELECTION

MAC Write/ZERO To G	MAC Write/Q To G	G Data Bits 52-63
0	0	P Mux Data bits 52-63
0	1	Q Mux 1 bits 52-63
1	0	0000 ₈
1	1	MAC Data bits 52-63

P RANK 1-8 REGISTER CHIP 1,2 MACROCELL

The P rank 1 through 8 register chip macrocell has the same chip type as the A, R, and Q registers' chip macrocell (CT 3.3). Refer to A Rank 1-8 Register Chip 1, 2 Macrocell for description.

K BARREL/REGISTER

Size: 7 data bits

Bit Numbering 48 52 57
 | | |

The K register (BAS 2.6) holds the instruction opcode for the executing instruction. The K register is forced to all ones when a PP idles (1077 instruction). This register is for displaying the opcode on maintenance display or for storing the opcode in the status register.

K BARREL SELECT MUX

K Barrel Select bits 0 and 1 select one of the five inputs to the K barrel select mux (BAS 2.6).

Dump Mode (EC register bit 52) and a selection code of one selects 073₈. Table II-3-6 shows the selection.

TABLE II-3-6. K BARREL SELECT MUX SELECTION

K Barrel	Select Bit	Dump Mode	K Barrel Select Bits 48, 52-57
0	1		
0	0	-	K Register Rank 0 bits 48, 52-57
0	1	Yes	073 ₈
0	1	No	071 ₈
1	0	-	PPM Data Rank 0 bits 48, 52-57
1	1	-	177 ₈

At deadstart, Deadstart Rank 0 forces K Barrel Select bit 1 to be active. This enables the K barrel select mux to select 071 which is the opcode for a block channel input instruction.

K RANK 1-8 REGISTER CHIP 1, 2 MACROCELL

Barrel chip address gen 1 (BAS 2.7) is a counter that counts from one through eight. The count bits generated from this counter are Barrel Chip Gen 1 bits 0 through 2. These bits are the Output/Write Enable and Read/Write Address bits 0 and 1 of the macrocell (CT 3.4). The K register shares the same two macrocell chips with the I register. Refer to A Rank 1-8 Register Chip 1, 2 Macrocell for description.

I REGISTER/BARREL

Size: 9 data bits plus 1 parity bit

Bit Numbering 52 63

The I register (BAS 2.7) holds the ROM address for the micrand field. Firmware stored in the microcode ROM (BAS 2.8) controls the execution of instructions. Refer to appendix C for definition of the micrand fields.

The opcode (PPM Data Rank 7 bits 48, 52 through 57) enters the I register through the I barrel select mux (BAS 2.7) when New Function is active. I Barrel Select bits 55 through 63 are ORed with Idle From C Macro Rank 7. The data clocks through I rank 8 register and addresses the microcode ROM.

The microcode ROM outputs the micrand fields which control the execution of all hardware. I Register Rank 9 bits 55 through 63 address the branch 1 and 2 ROM (BAS 2.7). The outputs, Branch 1 bits 49 through 60 and Branch 2 bits 61 through 72 enter the branch select mux (BAS 2.7). Branch select mux selects one of the five inputs, depending on Branch Select bits 0 and 1 and Dump Mode. Branch Select bits 0 and 1 are determined by the conditions that enter the microcode control decode macrocell (BAS 2.8).

BRANCH SELECT MUX

Table II-3-7 shows the selection of the branch select mux.

TABLE II-3-7. BRANCH SELECT MUX SELECTION

Branch	Select Bit	Dump Mode	Branch Select Bits 55-63
0	1		
0	0	-	Branch 2 bits 61-69
0	1	-	I Register Rank 9 bits 55-63
1	0	-	Branch 1 bits 49-57
1	1	Yes	775 ₈
1	1	No	776 ₈ (Microcode address of the first micrand)

Branch Select bits 0 and 1 are outputs of branch control A and B muxes respectively (CE 3.0). Branch Condition Select bits 59 through 63 control the selection of these muxes.

MICROCODE CONTROL DECODE MACROCELL

This macrocell (CE 3.0) contains the translation logic for controlling the PP registers during implementation of PP instructions. The macrocell has a channel and CM status translator and eight conditions muxes.

Channel and CM Status Translator

The channel and CM status translator (CE 3.0) translates channel and CM status functions selected by the eight condition muxes. For example, A to A mux has eight inputs. Each input is a condition that has to be met for A to A to be active. Micrand bits 24 through 26 select one of the eight inputs.

Condition Muxes

The eight condition muxes are: A to A, Q to Q, PPM write, exit, P to P, G control, branch control A, and branch control B (CE 3.0). Each mux selects a particular condition for that particular control to be active. Micrand Rank 9

bits 24 through 43 and Branch Condition Select bits 59 through 63 (BAS 2.8) control the selection of these muxes.

I RANK 0-7 REGISTER CHIP 1.2 MACROCELL

The I register shares the same two macrocell chips with the K register. Barrel Chip Gen 1 bits 0 through 2 are the Output/Write Enable and Read/Write Address bits 0 and 1 of the macrocell (CT 3.5). They are generated by the barrel chip address gen 1 (BAS 2.7) which counts from one to eight. Refer to A Rank 1-8 Register Chip 1, 2 Macrocell for description.

C BARREL/REGISTER

Size: 12 data bits plus 3 parity bits

Bit Numbering 52 63
|-----|

The C register (BAS 2.9) is used during 12/16 bit I/O conversion instructions (1071 and 1073). It holds the least significant 12 bits of the previous data word being transferred from channel or PPM, depending on the instruction.

The three parity bits in C register are generated on a four-bit boundary. These three parity bits are checked against the original parity of the data word.

C INPUT SELECT MUX

The C input select mux (BAS 2.9) selects either PPM or channel data or C register rank 0 data. Channel to C and FD to C are the select bits which are translated by Barrel 0 Channel Full, Active, and Micrand Rank 9 bits 82 through 84. Table II-3-8 shows the selection.

TABLE II-3-8. C INPUT SELECT MUX SELECTION

Channel To C	FD To C	C Input Select Bits 52-63
0	0	C Register Rank 0 bits 52-63
0	1	PPM Data Rank 0 bits 52-63
1	0	Barrel 0 Channel Data bits 52-63

CHANNEL INPUT/OUTPUT CONVERSION MACROCELL

This conversion macrocell (CB 3.0) performs 16-bit to 12-bit conversion on data from PPM to channels and vice-versa on data from channels to PPM. It also selects a 16-bit to 16-bit path for data in either direction.

Two chips are required: one for input conversion mode, the other for output conversion mode, but only one chip type is used. Output mode (CB 3.0) determines the mode of the chip (logic one for output mode; logic zero for input mode).

Output data can be forced to zero (parity bit to one) if Micrand Rank 8 bit 78 is zero. The parity bit is inverted if Micrand Rank 8 bit 78 is one and Invert Parity on Data to Channel Rank 8 is zero. A zero on Micrand bit 78 overrides the invert parity signal.

Micrand bits 79 through 81 control the conversion functions. Two input data paths are: a 15-bit input path (12 data bits plus 3 parity bits) from the C register, and a 17-bit input path (16 data bits plus 1 parity bit) from channel or A/PPM data. Input data is checked for odd parity and Input/Output Conversion PE indicates the parity error. Input/Output Macrocells P1 through P3 provide the required parity bits for the C register.

The three control signals, Micrand bits 79 through 81, are translated into mux select bits for each four-bit conversion mux. Table II-3-9 shows the selection of the four conversion muxes and the parity mux. Bit numbers followed by a C (that is, C bits 52 through 55) indicate C register data. Bit numbers followed by an I indicate either A/PPM data for the output macrocell or channel data for the input macrocell.

TABLE II-3-9. CONVERSION MUXES AND PARITY MUX SELECTION (1 of 2)

Micrand Bits			Output Mode	Operation	Mux Select Codes				
79	80	81			48-51	52-55	56-59	60-63	Parity
0	0	0	0	12/16 WORD 1	0	0	0	0	0
0	0	1	0	12/16 WORD 2	1	1	1	1	1
0	1	0	0	12/16 WORD 3	2	2	2	2	2
0	1	1	0	NOT USED	X	X	X	X	X
1	0	0	0	NOT USED	X	X	X	X	X
1	0	1	0	NOT USED	X	X	X	X	X
1	1	0	0	NOT USED	X	X	X	X	X
1	1	1	0	16/16	4	2	2	2	7
0	0	0	1	16/12 WORD 2	X(*)	1	3	0	0
0	0	1	1	16/12 WORD 1	X(*)	3	1	1	1
0	1	0	1	16/12 WORD 4	X(*)	4	4	4	2
0	1	1	1	16/12 WORD 3	X(*)	0	0	3	3
1	0	0	1	NOT USED	X(*)	X(*)	X	X	X
1	0	1	1	NOT USED	X(*)	X(*)	X	X	X
1	1	0	1	NOT USED	X	X	X	X	X
1	1	1	0	16/16	4	2	2	2	7
X Don't Care									
* Zeros are forced at output of muxes									

TABLE II-3-9. CONVERSION MUXES AND PARITY MUX SELECTION (2 of 2)

Mux SEL Code	Conversion Data Bit				Conversion Parity Bits							
	48-51	52-55	56-59	60-63	Input Mode				Output Mode			
0	C Bits	C Bits	C Bits	I Bits	CP1	CP2	CP3	IP1	CP3	IP0	IP1	
1	52-55	56-59	60-63	52-55	CP2	CP3	IP1	IP2	IP0	IP1	IP2	
2	C Bits	I Bits	I Bits	I Bits	CP3	IP1	IP2	IP3	CP1	CP2	CP3	
3	60-63	52-55	56-59	60-63	CP2	CP3	IP2	IP3	CP2	CP3	IP0	
4	C Bits	I Bits	I Bits	I Bits	NOT USED				IP0 IP1			
5	I Bits	C Bits	C Bits	C Bits					CP2 CP3			
6	48-51	52-55	56-59	60-63	NOT USED				NOT USED			
7	NOT USED	NOT USED	NOT USED	NOT USED	IP0	IP1	IP2	IP3	IP0	IP1	IP2	IP3

CP1 - Parity of C bits 52-55
CP2 - Parity of C bits 56-59
CP3 - Parity of C bits 60-63
IP0 - Parity of I bits 48-51
IP1 - Parity of I bits 52-55
IP2 - Parity of I bits 56-59
IP3 - Parity of I bits 60-63

The Converted Data Bit Parity on CB 3.0 is the generated parity of Input/Output Mode P0 through 3 in table II-3-9. Note that in output mode for conditions 4 and 5 (Micrand bits 79 through 81 equals 100 and 101), Conversion Data Bits 52 through 55 are zeros. These zeros are forced with emitter ANDs on the output of the conversion muxes.

Data integrity is maintained through the chip by generating a four-bit boundary parity on the input data and checking the generated parity bits against the input parity bit. The four parity bits are combined according to the control code to form one output parity bit for the 16-bit data word.

C REGISTER CHIP 1.2 MACROCELL

This C register macrocell (CT 3.6) has two chips: chip 1 and 2. Chip 1 shifts C register bits 54 through 63 from rank 1 to rank 8. Chip 2 shifts C register bits 52 and 53 from rank 1 to rank 7 and C parity bits 1 through 3 from rank 2 to rank 8. Barrel chip address gen 1 (BAS 2.7) counts from 1 to 8 and provides Barrel Chip Gen 1 bits 0 through 2 for chip 1. Barrel chip address gen 2 (BAS 2.9) counts from 2 to 8 and provides Barrel Chip Gen 2 bits 0 through 2 for chip 2. Refer to A Rank 1-8 Register Chip 1, 2 Macrocell for description.

SIGNAL FLOW

The load d instruction (0014 d) is used to illustrate signal flow through the barrels. A flowchart illustrating signal flow is shown in appendix B.

The instruction is read from PPM (PPM 2.1). PPM Data Rank 7 bits 48, 52 through 57 (opcode) enters the I barrel (BAS 2.7) through the I barrel select mux. New Function selects the PPM data. New Function originates from New Function From Microcode Macrocell Rank 9 (BAS 2.8) which is the output of exit mux (CE 3.0). Exit mux selects one of the eight exit conditions (new functions) as determined by Micrand Rank 9 bits 38 through 40.

If Idle From C Macrocell Rank 7 (BAS 2.7) is not set, PPM Data Rank 7 bits 48, 52 through 57 clock through I Rank 8 register to address the microcode ROM (BAS 2.8). This ROM contains the micrand bits for the 0014 instructions.

The d field (PPM Data Rank 9 bits 58 through 63) enters the A barrel through the A adder B operand mux (BAS 2.1). D to A Rank 9 (Micrand bit 6) selects the d field to be the B input of the A adder. The A Adder Code bits 0 through 4 Rank 9 decode a value of five for the B input (d field) to be A Adder Result bits 58 through 63. A Adder Result bits 46 through 57 are zeros. After clocking through A rank 0 register (BAS 2.1), A Register Rank 0 bits 46 through 63 enter the A to A mux (BAS 2.0). A to A Rank 0 being not active enables A Register Rank 0 bits 46 through 63 to enter the A shifter macrocell (BAS 2.0). No shift occurs in the shifter macrocell because Shift Rank 8 (micrand bit 3) is not active.

During the instruction, the microcode loads the d field into the A register. It also increments P register and sets exit condition (Micrand Rank 9 bits 38 through 40 equals 110) which produces the next New Function signal.

PERIPHERAL PROCESSOR MEMORY

The peripheral processor memory (PPM) stores a maximum of 4096 words (one word has 16 data bits and 1 parity bit). Each memory pak (CM) contains 10 PPMs. PPM Address bits 52 through 63 enter the bank 0 through 2 rank 1 address register (PPM2.0) where they are strobed by Bank 0 through 2 Strobes. These address bits are converted to TTL levels before they address the PPM. Input data to the PPM are Y Mux Data bits 48 through 63 (PPM 2.1) from the Y mux (PPM 2.2). These data bits enter the bank 0 through 2 transceiver where they are strobed by Bank 0 through 2 Strobes (PPM 2.1) and converted to TTL levels before entering the PPM.

R bits 60 through 63 (PPM 2.0) reconfigure the PPs at deadstart time. They are loaded into the PPM counter rank 0 at MCC=9. The counter counts from 0 through 11g. The output of the counter, C bits 60 through 63, are compared with 1001g. If a count of 11g is reached, Count=11 from the load counter comparator reloads the counter with zeros.

The count bits, C bits 60 through 63, are decoded by the bank active decoder (PPM 2.0). Bank 0 is active for a decode of 0, 3, 5, or 10; bank 1 for a

decode of 1, 6, or 11 and bank 2 for a decode of 2, 4, or 7. Banks 0 through 2 Active provide the strobe, write enable, and output disable signals for the PPM.

PPM DATA RANK 3-6 MACROCELL

This macrocell (CT 3.7) shifts the PPM data from rank 3 to rank 6. Address bits 0 and 1 (PPM 2.1) from read/write address counter which counts from 0 to 3 provide Read/Write Address bits 0, 1 for the macrocell. Refer to A Rank 1-8 Register Chip 1, 2 Macrocell for description.

Y MUX

The Y mux (PPM2.2) selects input write data to the PPM according to Y Mux Select bits 62, 63 if ADU Write is not active. The input write data consists of one of the following:

- A Bypass Rank 0 bits 48 through 63
- MAC Input to Y bits 48 through 51, MAC/RA Mux bits 52 through 63
- Channel Data to A/Y bits 48 through 63
- P Register Rank 0 bits 52 through 63

The Y mux also disassembles ADU data according to Word Count 1, 2 Rank 0 if ADU Write is active. The output of the Y mux, Y Mux Data bits 48 through 63, is strobed and converted to TTL levels before it writes into PPM.

Y Mux Data Parity passes through PPM data in generator/checker (PPM 2.2) if 16 Bit Mode Rank 0 is active. Parity is generated for all Y mux inputs except P register data if 16-Bit Mode Rank 0 is not active. Regenerated P Parity does not depend on 16-Bit Mode Rank 0. Data parity is checked at the output of Y mux.

SECTION II-4

INPUT/OUTPUT CHANNELS

=====

This section describes the operation of input/output (I/O) channels (0, 1, 3, 4, 6, 7, 11, 12, 20, 21, 23, 24, 26, 27, 31, and 32) only. Refer to sections 6 and 7 of this part for description of channels 10 and 15 and part III for description of channel 17. The I/O channels provide the only data paths between the mainframe and external devices. There is no direct central memory (CM) access.

The following I/O instructions can be executed on the I/O channels for data communication and status checks. For more information refer to appendix B.

<u>Function</u>	<u>Code</u>	<u>Description</u>
Channel Branch Conditions	00640cm	Jump to m if channel c active
	00641cm	Jump if channel c flag set and set flag if clear
	00650cm	Jump to m if channel c inactive
	00651cm	Clear flag on channel c and jump to P+2
	00660cm	Jump to m if channel c full
	00661cm	Jump to m if channel c error flag set
	00670cm	Jump to m if channel c empty
Data Transfer	00671cm	Jump to m if channel c error flag clear
	00700c	Input to A from channel c when active and full
	00701c	Input to A from channel c if active and full
	0071Xcm	Input (A) words to m from channel c
	00720c	Output from A on channel c when active and empty
	00721c	Output from A on channel c if active and empty
	0073Xcm	Output (A) words from m on channel c
Channel Control	00740c	Activate channel c when inactive
	00741c	Unconditionally activate channel c
	00750c	Deactivate channel c when active
	00751c	Unconditionally deactivate channel c
Device Protocol	00760c	Function (A) on channel c when inactive
	00761c	Function (A) on channel c if channel inactive
	00770cm	Function m on channel c when inactive
	00771cm	Function m on channel c if inactive

The channel instructions are represented in one of two formats. The first format uses a single 16-bit word; the second uses two consecutive 16-bit words.

16-bit format	4	5	5	6
	8	2	8	3
	g	e	f	s c

32-bit format

4	5	5	6
8	2	8	3
g	e	f	s
0	0	0	0
		m	

The following field descriptions apply to both instruction formats:

- g 1 bit Most significant bit of 7-bit operation code. In many instructions, g acts to control width of operand read from PPM. If g is 0, operand is 12 bits; if g is 1, operand is 16 bits.
- e 3 bits Not used. Reserved for future use; must be set to zero.
- f 6 bits Least significant 6 bits of 7-bit operation code.
- s 1 bit A sub-operation code used with certain I/O instructions.
- c 5 bits A channel number.
- 0000 Not used, must be set to zero.
- m 12 bits Part of an operand, an address specification, or an I/O function code depending upon the instruction.

CONFIGURATION

Maximum configuration comprises 4 special channels, 7 internal channels, 6 ICI channels and 12 CYBER 170 channels. Table II-4-1 shows the channel allocation.

Each CH channel pak consists of six channels: one integrated controller interface (ICI) channel, two internal channels and three CYBER 170 channels. The DH channel pak consists of six channels: three integrated controller interface (ICI) channels, two internal channels and one CYBER 170 channel. All channels except channel 14 can be used for inter-PP communication which is always on a 16-bit boundary. IOU supports CYBER 170 peripheral equipment using CYBER 170 channel. Communication is on a 12-bit boundary. IOU also converts 12-bit channel words to 16-bit PP words and 16-bit PP words to 12-bit channel words through special I/O instructions.

CYBER 170 CHANNELS

Channels 4, 7, 11, 12, 21, 23, 24, 27, 31, and 32 are CYBER 170 channels. Internal interface is between channel control and data macrocell (CHL 2.0, 2.1), and PP. External interface is between the channel control and data macrocell, and the external device.

The CH channel pak provides CYBER 170 channel numbers 1 and 3. Channels 1 and 3 become ICI channels in DH channel pak.

INTERNAL INTERFACE

Each channel internal interface consists of four control signals for each barrel (CHL 2.0). The four control signals are: Full, Active, Flag, and

TABLE II-4-1. CHANNEL ALLOCATION

Channel Number	Channel Type
1 0	ICI) I/O Channel
2 1	ICI or CYBER 170) I/O Channel
3 2	Internal*
4 3	ICI or CYBER 170) I/O Channel
5 4	CYBER 170) I/O Channel
6 5	Internal
7 6	ICI) I/O Channel
8 7	CYBER 170) I/O Channel
9 10	DSC Special Channel
10 11	CYBER 170) I/O Channel
11 12	CYBER 170) I/O Channel
12 13	Internal
13 14	RTC) Special Channel
14 15	TPM) Special Channel
15 17	MCH) Special Channel
16 20	ICI) I/O Channel
17 21	CYBER 170) I/O Channel
18 22	Internal *
19 23	CYBER 170) I/O Channel
20 24	CYBER 170) I/O Channel
21 25	Internal
22 26	ICI) I/O Channel
23 27	CYBER 170) I/O Channel
24 30	Internal
25 31	CYBER 170) I/O Channel
26 32	CYBER 170) I/O Channel
27 33	Internal

16?

→ ☐ Cyber 170 Channels

Error Flag. The internal interface also consists of a 16-bit bidirectional data bus for each barrel (CHL 2.1).

Full (CHL 2.0)

Full is set (channel full) by a PP using output instructions (00720, 00721, 0073X, 1073X) once for each word written, or by an external device whenever a data word is written into the channel mux register (CD 3.0).

Full is cleared by a PP using deactivate instructions (00740, 00741), at the end of input transfer instructions (00700, 00701, 00710, 00711), or by an external device after the external device recognizes the full condition and reads the channel mux register.

Full is sensed by a PP using full/empty jump instructions (00660, 00670).

Active (CHL 2.0)

Active is set (channel active) by a PP using activate instructions (00740, 00741), by function instructions (00760, 00761, 00770, 00771), or by an external device. Channel active indicates the channel is reserved for channel communication.

Active is cleared (channel inactive) by a PP using deactivate instructions (00750, 00751), or by an external device. Channel inactive indicates that channel communication is complete and that the channel may be used by other PPs. Normally, external devices clear the active bit only at the end of an input transfer or in response to a function instruction.

Active is sensed by a PP using active/inactive jump instructions (00640, 00650) allowing a PP to wait until the channel is inactive before proceeding.

Flag (CHL 2.0)

Flag is set or cleared by a PP using flag set/clear jump instructions (00641, 00651).

Flag is sensed by a PP using flag set/clear jump instructions (1064X, 1065X) giving the PP programmer more control of peripheral operation when two PPs are using the same device.

Error Flag (CHL 2.0)

Each channel generates and checks parity (1 bit per word) for all data transferred on the channel. The channel error flag is set when a parity error is detected in the data in the channel mux register.

The channel error flag is sensed by a PP using channel error flag test and clear instructions (00661 and 00671).

EXTERNAL INTERFACE

The 12-bit external interface provides synchronous communication between the IOU and CYBER 170 external devices. The transmission is accomplished via separate input and output cables. Each cable transmits 13-data signals (12 data bits plus 1 parity bit). Eight control signals are transmitted from a PP to an external device, and four from the external device to a PP. The signals are transmitted over coaxial cables using an AC transmission scheme.

Some external devices connected with the data and control lines relay all signals to the next-in-line device. The relay is synchronized with a 10-MHz clock which is one of the control signals. This synchronization causes all devices to be synchronous with the IOU but displaced from each other by one or more 100-nanosecond clock periods. Since the signals are retransmitted at each device interface, powering off one device disables data transmission to all devices beyond. The maximum cable length between devices is 21 meters (70 feet).

A 12-bit external interface transmits data between bits 52 through 63 of the interface channel and the external device. On output, bits 48 through 51 of the internal channel word are not transmitted; on input, bits 48 through 51 of the internal channel word are cleared.

The external control signals are:

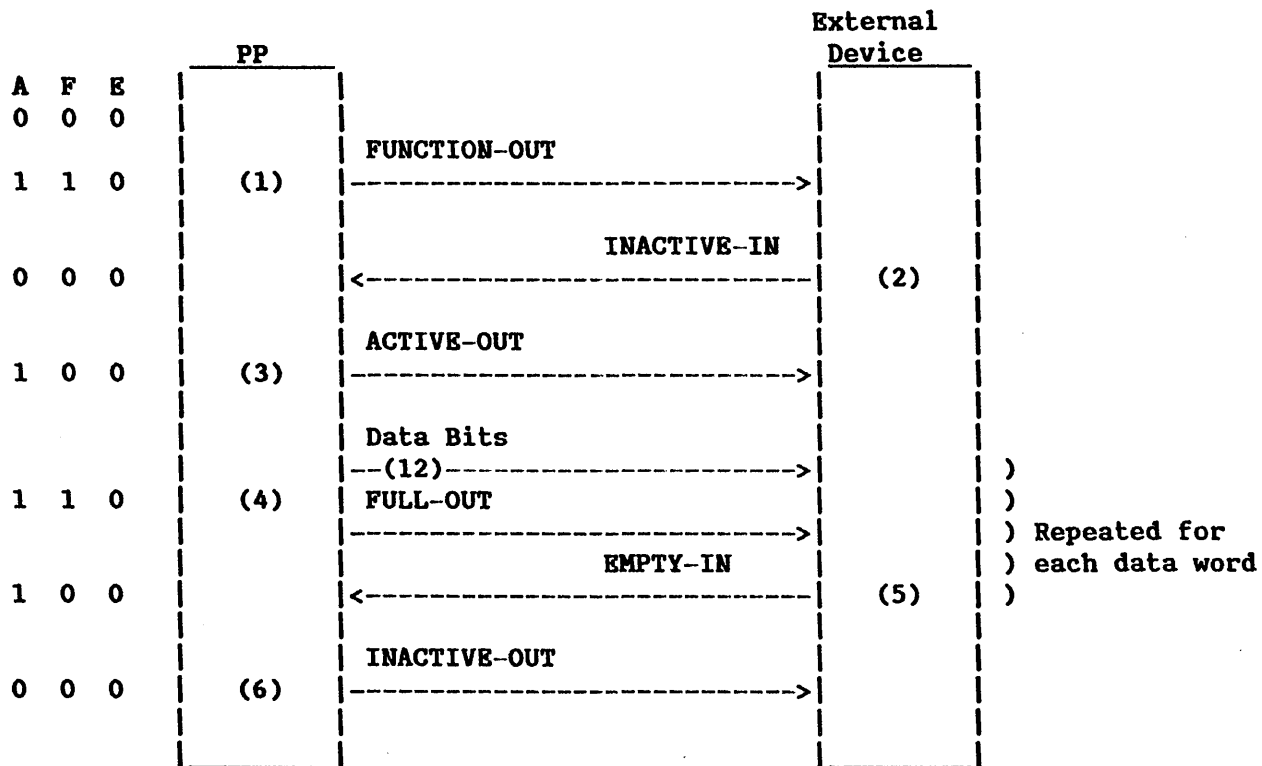
ACTIVE	Sent normally by a PP to an external device; indicates beginning of data transmission. Active-Out and Active-In are transmitted/received on CHL 2.0.
INACTIVE	Sent from data sending or data receiving device; signifies end of data transmission and clears active and full flags. Inactive-Out and Inactive-In are transmitted/received on CHL 2.0.
FULL	Sent from data sending device to data receiving device at the same time that data is transmitted. Full indicates that receiving device must sample the data signals. Full-Out and Full-In are transmitted/received on CHL 2.0.
EMPTY	Sent by data receiving device to data sending device to acknowledge receipt of a full pulse and associated data. Empty indicates to sending device that new data may be transmitted. Empty-Out and Empty-In are transmitted/received on CHL 2.0.
FUNCTION	Sent to external device to indicate that associated signals are control signals not data. Function-Out is transmitted on CHL 2.0.
MASTER CLEAR	Sent by IOU to all external devices on I/O channel to cease all activity and restore initial conditions. Master Clear is transmitted on CHL 2.2.
10-MHZ CLOCK	Sent by IOU to external device; consists of a pulse sent every 100 nanoseconds to synchronize all external devices with IOU. The clock is transmitted on CHL 2.2.
1-MHZ CLOCK	Sent by IOU to external device; consists of a pulse sent every 1 microsecond. The clock is transmitted on CHL 2.2.

CHANNEL PROTOCOL

Data Output Sequence

Figure II-4-1 shows the data output sequence. The number preceeding each paragraph below corresponds to the number in figure II-4-1.

- (1) PP executes a function instruction which sets the active and full bits in the internal interface, places a function word in the channel register (CD 3.0), and sends a function-out pulse to accompany the function word on the channel lines (CC 3.0).



Key

A F E : Active, Full, and Error bits

Figure II-4-1. Data Output Sequence

The 00760, 00761, 00770, and 00771 instructions result in a function pulse. The last digit (0 or 1) sets the escape condition. If not set (0) the PP hangs until the desired conditions are met. The instruction is read from PPM, the op code enters the I barrel (BAS 2.7) and addresses the microcode ROM (BAS 2.8). The microcode ROM determines the conditions of Micrand Rank 8 bits 73 through 75, 78, and 85. Q Register Rank 8 bits 59 through 63 (BAS 2.4) select the channel.

The RWCH field (Micrand Rank 8 bits 73 through 75) is decoded by barrel 0 and barrel 1 channel function translator (CC 3.0). The resulting channel functions are shown in table II-4-2. The channel function is delayed, resynchronized, and sent to the control transmitter (CHL 2.0). The transmitter sends the control signal Function-Out over the channel to the external device. The MCH field (micrand bits 79 through 83) controls the channel word.

- (2) The external device acknowledges the acceptance of the function by sending an Inactive-In signal (CHL 2.0). This signal drops the active flag and the full flag (CC 3.0), and clears the channel register (CD 3.0).

TABLE II-4-2. RWCH BITS 73 THROUGH 75

Code	Function
0	FUNCTION
1	ACTIVE
2	FULL
3	EMPTY
4	INACTIVE
5	CLEAR ERROR
6	CLEAR FLAG
7	DO NOTHING

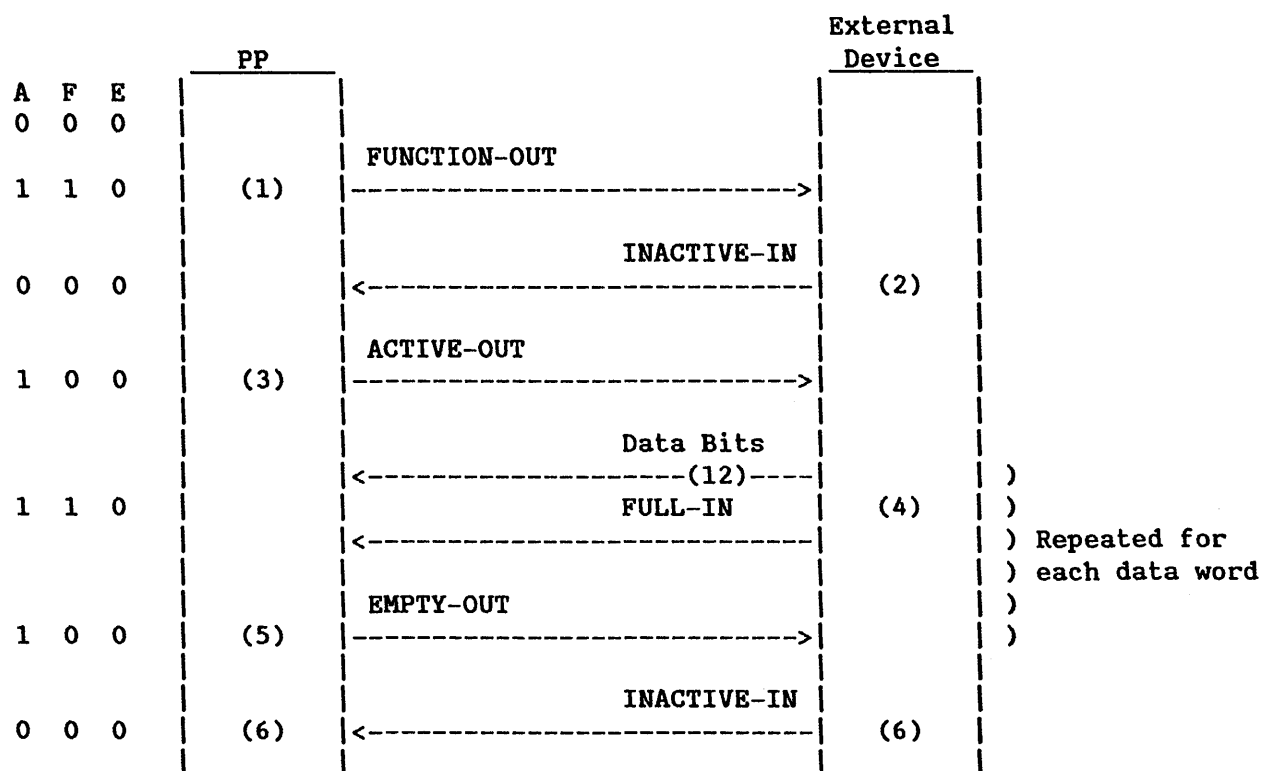
The Inactive-In signal from the external device is received by the external receiver (CHL 2.0).

- (3) PP sends Active-Out to indicate that data flow is about to start (CHL 2.0).
- (4) PP places a word (12 data bits plus 1 parity bit) in the channel register to set the full flag and sends a Full-Out signal (CHL 2.0).
- (5) The external device accepts the data word and sends an Empty-In signal to clear the channel register and the full flag (CHL 2.0).
- (6) After steps (4) and (5) have been repeated to complete the data transfer, PP sends an Inactive-Out signal to turn off the external device (CHL 2.0).

Data Input Sequence

Figure II-4-2 shows the data input sequence. The number preceeding each paragraph corresponds to the number in figure II-4-2.

- (1) PP executes a function instruction which sets the active and full bits in the internal interface, places a function word in the channel register and on the channel lines, and sends a function pulse to accompany the function word (CHL 2.0).
- (2) The external device acknowledges the acceptance of the function by sending an Inactive-In signal (CHL 2.0). This signal drops the active flag and the full flag (CC 3.0), and clears the channel register (CD 3.0).
- (3) PP sends Active-Out to indicate that data flow is about to start (CHL 2.0).
- (4) The external device sends a 12-bit word plus parity to the channel register (CD 3.0) along with a Full-In signal (CHL 2.0) which sets the full flag.
- (5) PP stores the data word and clears the full flag which sends an Empty-Out signal (CHL 2.0) to the external device to acknowledge acceptance.



Key

A F E : Active, Full and Error bits

Figure II-4-2. Data Input Sequence

(6) After steps (4) and (5) have been repeated to complete the data transfer, the external device clears its active condition and sends an Inactive-In signal (CHL 2.0) to the PP to clear the channel active flag.

ICI CHANNELS

Channels 0, 6, 20, and 26 are ICI channels. If DH pak is installed, channels 1 and 3 are also ICI channels. ICI is an interface between the I/O channels and device controllers which are housed inside the IOU cabinet. These device controllers, called integrated controllers, share the backpanel and power supplies of IOU. IOU and integrated controllers are in the same environmental monitor control. ICI uses synchronous communication between the IOU and integrated controller.

INTERNAL INTERFACE

Each ICI channel internal interface consists of four control signals for each barrel (CHL 2.0). They are Full, Active, Flag, and Error Flag. Their descriptions are identical to those of the four control signals of the CYBER 170 channels.

EXTERNAL INTERFACE

External interface consists of nine control signals, one Master Clear signal, and 17 bidirectional data/parity signals. Figure II-4-3 shows the ICI.

Control Signals

All control signals are 50 ± 2 -ns pulses, emitter-coupled logic (ECL) levels, and active high. The nine control signals are Function-Out, Full-Out, Active-Out, Empty-Out, Inactive-Out, Full-In, Empty-In, Inactive-In, and Error-In. All control signals except Error-In are described in the CYBER 170 Channel section. Error-In (CHL 2.0) is sent by the integrated controller to the CC control macrocell if there is an error in the integrated controller. Error-In sets the channel error flag (CC 3.0).

Master Clear

The Master Clear signal (MC Delay on CHL 2.2) is sent to the integrated controller. The signal is active high and its pulse width is a minimum of 1 microsecond and a maximum of 100 milliseconds. It is generated from the power-on master clear of the -5.2V supply or the IOU deadstart master clear.

Data/Parity

The 16/1 bidirectional data/parity lines are ECL levels. Data lines are active low. Parity is odd-parity. If the controller is not connected, parity checking is valid at the controller side only during Function-Out. If the controller is connected, parity checking is valid during Function-Out or Full-Out.

ICI TIMING

ICI uses the IOU system clock, T1. T1 is a 20 ± 0.4 -MHz clock with a 10 ± 2.5 -ns pulse width. The maximum clock skew of T1 signals measured at the connector pin of a channel pak and the connector pin of an integrated controller pak is ± 2.0 -ns. The interface signal timing is defined at the connector pins of the IOU channel pak. Figure II-4-4 shows the ICI timing.

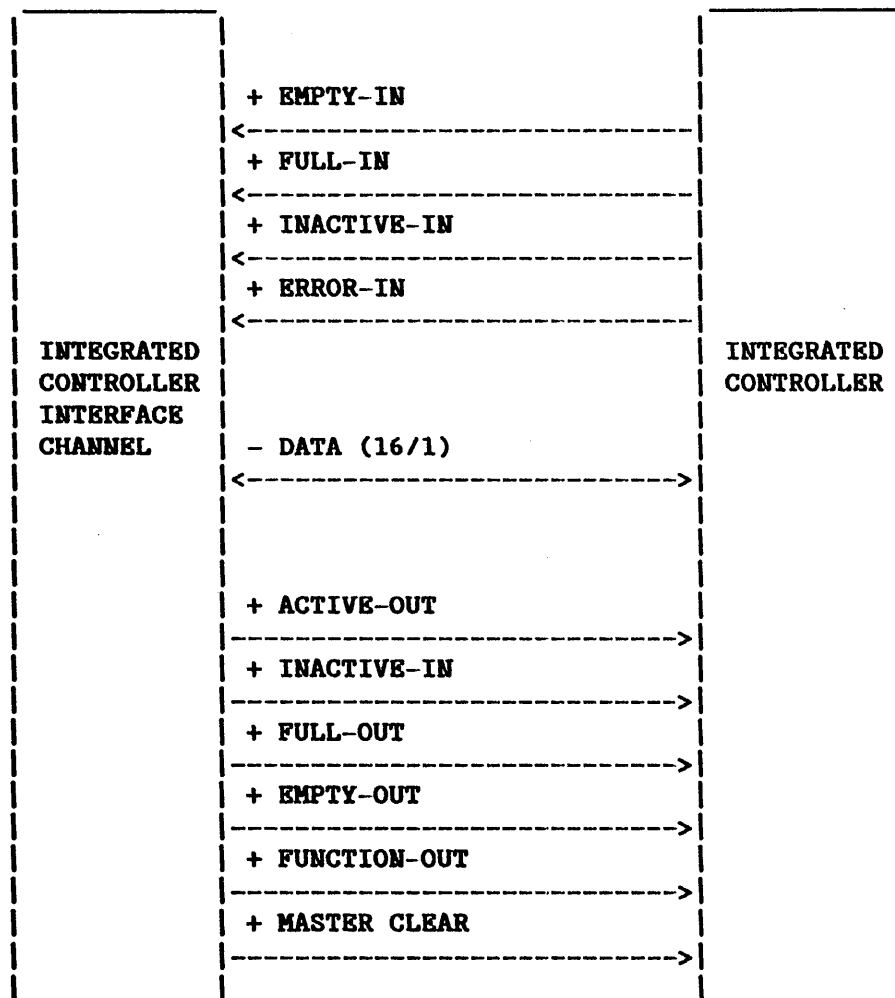


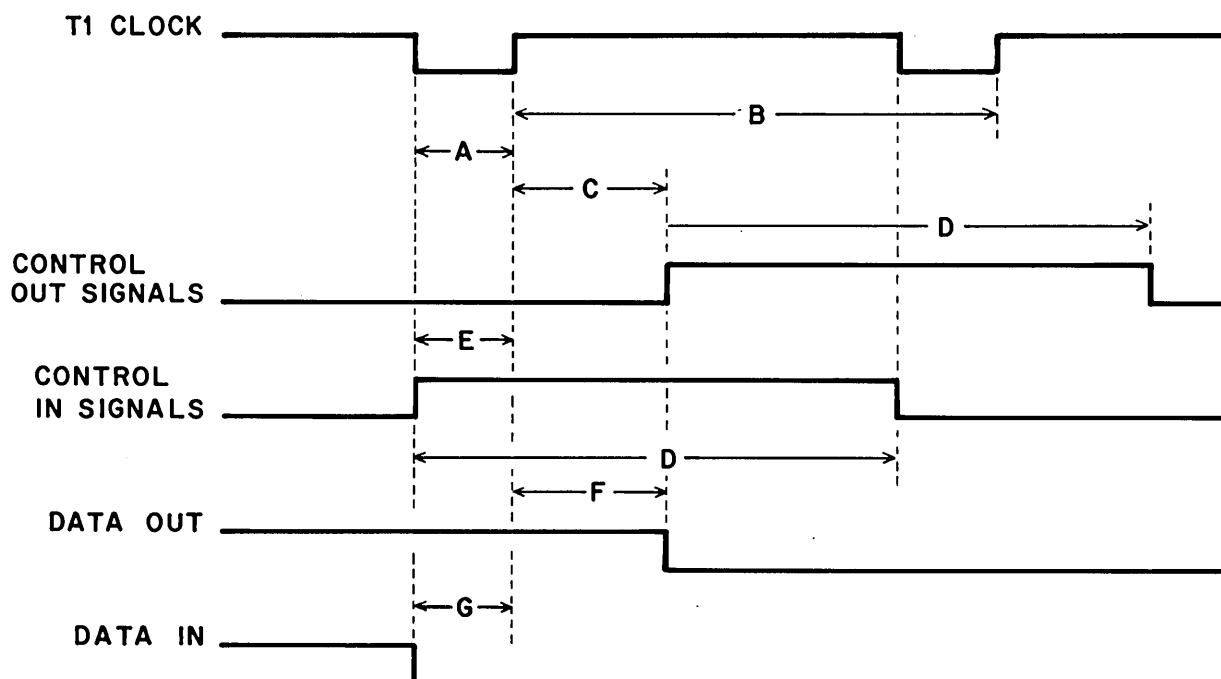
Figure II-4-3. Integrated Controller Interface (ICI)

DATA/PARITY TRANSMISSION SCHEME

Data transmission from the channel to the integrated controller (CHL 2.1) occurs when the PP sends a Function-Out or Full-Out to the integrated controller (CHL 2.0). Transmission continues until the integrated controller sends an Inactive-In or the PP issues an Inactive-Out. MC Delay (CHL 2.1) also disables the transmission.

Data transmission from the integrated controller to the channel occurs when the integrated controller sends a Full-In to the channel. Transmission discontinues when the integrated controller receives an Inactive-Out or issues an Inactive-In. MC Delay also disables the transmission.

The channel then outputs a Full-Out or a Function-Out and data at the same time. The integrated controller must compensate for any skew at the controller and between these signals due to different cable lengths.



CI085

A - T1 clock pulse width	= $10 \pm 2/5$ ns
B - T1 clock period	= 50 ± 1 ns
C - Control signal transmit time	= Minimum 3.0 ns Maximum 15.0 ns
D - Control signal pulse width	= 50 ± 2 ns
E - Control signal setup time	= Minimum 9.0 ns
F - Data transmit time	= Minimum 5.0 ns Maximum 20.0 ns
G - Data setup time	= Minimum 2.0 ns

Figure II-4-4. ICI Timing.

When the integrated controller is not selected, it does not interfere with the operation of the channel and it ignores all data and control signals except Function-Out (CHL 2.0) and MC Delay (CHL 2.2).

CHANNEL CONTROL MACROCELL

The control macrocell (CC 3.0) contains the logic necessary to implement all the control operations for one I/O channel. Each CH channel pak has six control macrocells, one for each channel. For channels 1, 3, and 4, T20 clocks the CYBER 170 channel controls through the 170 Mode Register and the external control mux selects these controls. For channel 0, the external control mux selects the inputs which do not pass through the 170 Mode Register. The channel status translator (CC 3.0) consists of the active and full flip-flops where Active and Full are serially shifted through the MAC serial data bus (CHL 2.0) into MAC. The function translator (CC 3.0) consists of channel flag and error flip-flops where Flag and Error are also serially shifted into MAC. The external control mux circuit (CC 3.0) is the interface between the PPs and the channel.

CHANNEL DATA MACROCELL

The data macrocell (CD 3.0) is an interface between channel and PP data. Each macrocell controls data flow for three channels. Data path is five bits wide. There are three channel mux registers, one for each channel. Barrel 0 or 1 can read/write each of the three registers. In addition, both barrels can read/write any two of the three registers or read the same register at the same time. However, they cannot write the same register at the same time.

The channel to barrel data mux selects channel 0, 1, or 2 data from the channel mux registers. When Barrel 0 Channel 0/1/2 Empty is active (high), it enables Channel 0/1/2 Barrel 0 Data bits to go to PPs. When this signal is inactive (low), the output PP data signals are all low.

In CYBER 170 mode, T20 latches external data into the 170 Buffer Register. This latched data is copied into the channel mux register by T1 when Channel Load for that channel is active and Barrel Channel 0/1/2 Function or Full is inactive.

In CYBER 180 mode, external data is loaded the same way as in CYBER 170 mode except it bypasses the 170 Buffer Register. The external parity signal is active at all times and is used only in CYBER 170 mode.

The external data-out signals in CYBER 170 mode are 50-ns pulses transmitted by T30. These signals come from the channel mux register.

The external data-out signals in CYBER 180 mode are levels that reflect the contents of the channel mux register. The channel mux register changes on every T1.

SIGNAL DEFINITIONS

The signals entering and leaving the macrocell are defined below (CD 3.0).

Channel 0 External Data, Channel 1 Received Data.

The external data-in signals (five bits to each macrocell) come from external equipment. These data signals are true (active low) if the channel is in CYBER 170 mode and are 25-nanosecond wide pulses. They are latched into the 170 Buffer Register anytime between the trailing edge of T20 and the trailing edge of T10 (minus set up time of about five nanoseconds).

The data-in signals in CYBER 180 mode are true and they remain valid until the external device sends another data word or terminates the transfer. They are not 50-ns pulses. Channel 2 has no external data-in interface.

Barrel 0, 1 Channel Data

The Barrel 0 and 1 Channel Data signals (five bits from each barrel) are 50-ns pulses coming from the PPs.

Barrel 0, 1 Channel 0/1/2 Function Or Full

This signal is a 50-ns pulse that selects input PP data if Channel Select is low. It also provides the strobe for the channel mux register.

Channel 0-2 Select Barrel 0, 1 Rank 9

Each barrel has a channel select signal for each channel. This signal is a true 50-ns pulse from T1. It selects one of the three channel mux registers used for the Barrel Channel Function or Full and Barrel Channel Empty signals. It also selects the input data barrel during Barrel Channel Function or Full signal.

Master Clear Delay

The Master Clear Delay signal (active low) is a 50-ns pulse from T1. It synchronizes the clock circuit in the macrocell with a low to high transition.

Channel 0, 1 Load

The Channel Load signal is active low. It enables external data to be strobed into the channel mux register. When in CYBER 170 mode, the signal is set anytime after the trailing edge of T20 and the trailing edge of T10 (minus set up time). It is cleared at the leading edge of T20. In CYBER 180 mode, this signal is a 50-ns pulse and is active low.

The load external data function is active when MC Delay is inactive. When MC Delay is active, the load signal sets the mode of the channel to CYBER 170 mode if Channel Load is active or CYBER 180 mode if Channel Load is inactive.

Barrel 0, 1 Channel 0/1/2 Empty

The Barrel Channel Empty signal is a false (active high) pulse from T1. When this signal is high, it enables data from channel mux register to go to PPs. When this signal is low, Channel 0/1/2 Barrel Data are all low.

Channel 1 External P2

This parity signal is the odd parity of the five data bits in the 170 Buffer Register.

Channel 0-2 P3

The channel data parity signal is the odd parity of the data in the channel mux register. It is a true signal.

Channel 0, 1 Data

The Channel 0, 1 Data are the data-out signals (five bit for each channel) that go to the external device. In CYBER 170 mode, these signals are true 50-ns pulses from trailing edge of T30 to trailing edge of T10. In CYBER 180 mode, these signals are true and always indicate the contents of the channel mux register.

Channel 0/1/2 Barrel 0, 1 Data

The Channel 0/1/2 Barrel Data (five bits for each barrel) are the output data that go to the PPs. These signals are true 50-ns pulses from T1 if Barrel 0 and 1 Channel 0/1/2 Empty are high.

DATA I/O SEQUENCES

Table II-4-3 gives an example of data I/O sequences.

The activities in the table are summarized below.

- o Cycle 00-01 Channel 0 and 1 Load being active set channels 0 and 1 to CYBER 170 mode.
- o Cycle 02-03 Barrel 0 writes 25 into channel 1. Barrel 1 write 31 into channel 2.
- o Cycle 04-05 Barrel 1 reads 25 from channel 1. External device writes 13 into channel 0.
- o Cycle 06-07 Channel 1 sends 25 to external device. Barrel 0 reads 31 from channel 2.
- o Cycle 08-09 Barrel 0 write 04 to channel 0.
- o Cycle 10-11 Channel 0 sends 04 to external device. Barrel 1 reads 04 from channel 0.

TABLE II-4-3. EXAMPLE OF DATA I/O SEQUENCES

Cycle Number	00	01	02	03	04	05	06	07	08	09	10	11	12
T1	0	1	0	1	0	1	0	1	0	1	0	1	0
T2	1	0	1	0	1	0	1	0	1	0	1	0	1
T10	X	X	0	1	0	0	0	1	0	0	0	1	0
T20	X	X	0	0	1	0	0	0	1	0	0	0	1
T30	X	X	0	0	0	1	0	0	0	1	0	0	0
T40	X	X	1	0	0	0	1	0	0	0	1	0	0
Master Clear Delay	1	1	0	0	0	0	0	0	0	0	0	0	0
Barrel 0 Channel Data	37	37	25	25	37	37	31	31	04	04	37	37	37
Barrel 1 Channel Data	37	37	31	31	25	25	37	37	37	37	04	04	37
Barrel 0 Channel 0/1/2 Fctn or Full	0	0	1	1	0	0	0	0	1	1	0	0	0
Barrel 1 Channel 0/1/2 Fctn or Full	0	0	1	1	0	0	0	0	0	0	0	0	0
Channel 0 External Data	00	00	00	00	00	13	00	00	00	00	00	00	00
Channel 1 Received Data	00	00	00	00	00	00	00	00	00	00	00	00	00
Channel 0 Load	1	1	0	0	0	1	0	0	0	0	0	0	0
Channel 1 Load	1	1	0	0	0	0	0	0	0	0	0	0	0
Channel 0 Select Barrel 0	0	0	0	0	0	0	0	0	1	1	0	0	0
Channel 1 Select Barrel 0	0	0	1	1	0	0	0	0	0	0	0	0	0
Channel 2 Select Barrel 0	0	0	0	0	0	0	0	0	0	0	0	0	0
Channel 0 Select Barrel 1	0	0	0	0	0	0	0	0	0	0	1	1	0
Channel 1 Select Barrel 1	0	0	0	0	1	1	0	0	0	0	0	0	0
Channel 2 Select Barrel 1	0	0	1	1	0	0	1	1	0	0	0	0	0
Barrel 0 Channel 0/1/2 Empty	0	0	0	0	0	0	1	1	0	0	0	0	0
Barrel 1 Channel 0/1/2 Empty	0	0	0	0	1	1	0	0	0	0	1	1	0
Channel 0/1/2 Barrel 0 Data	37	37	37	37	37	37	31	31	37	37	37	37	37
Channel 0/1/2 Barrel 1 Data	37	37	37	37	25	25	37	37	37	37	04	04	37
Channel 0 External Data	00	00	00	00	00	00	00	00	00	00	04	04	00
Channel 0 Output Data	00	00	00	00	00	00	25	25	00	00	00	00	00
Channel 0 P3	1	1	1	1	1	1	0	0	0	0	0	0	0
Channel 1 P3	1	1	1	1	0	0	0	0	0	0	0	0	0
Channel 2 P3	1	1	1	1	0	0	0	0	0	0	0	0	0
Channel 1 External P2	1	1	1	1	1	1	1	1	1	1	1	1	1

MAC INTERFACE SHIFTER

The MAC interface shifter (CHL 2.0) shifts the channel controls (Active, Full, Flag, and Error) serially through the chan MAC serial data bus into MAC. Byte address bits 62, 63, and P select the particular channel. For more information, refer to Bit Serial Control under MAC Operation in section III-3.

SECTION II-5

IOU MAINTENANCE REGISTERS

MAINTENANCE REGISTERS

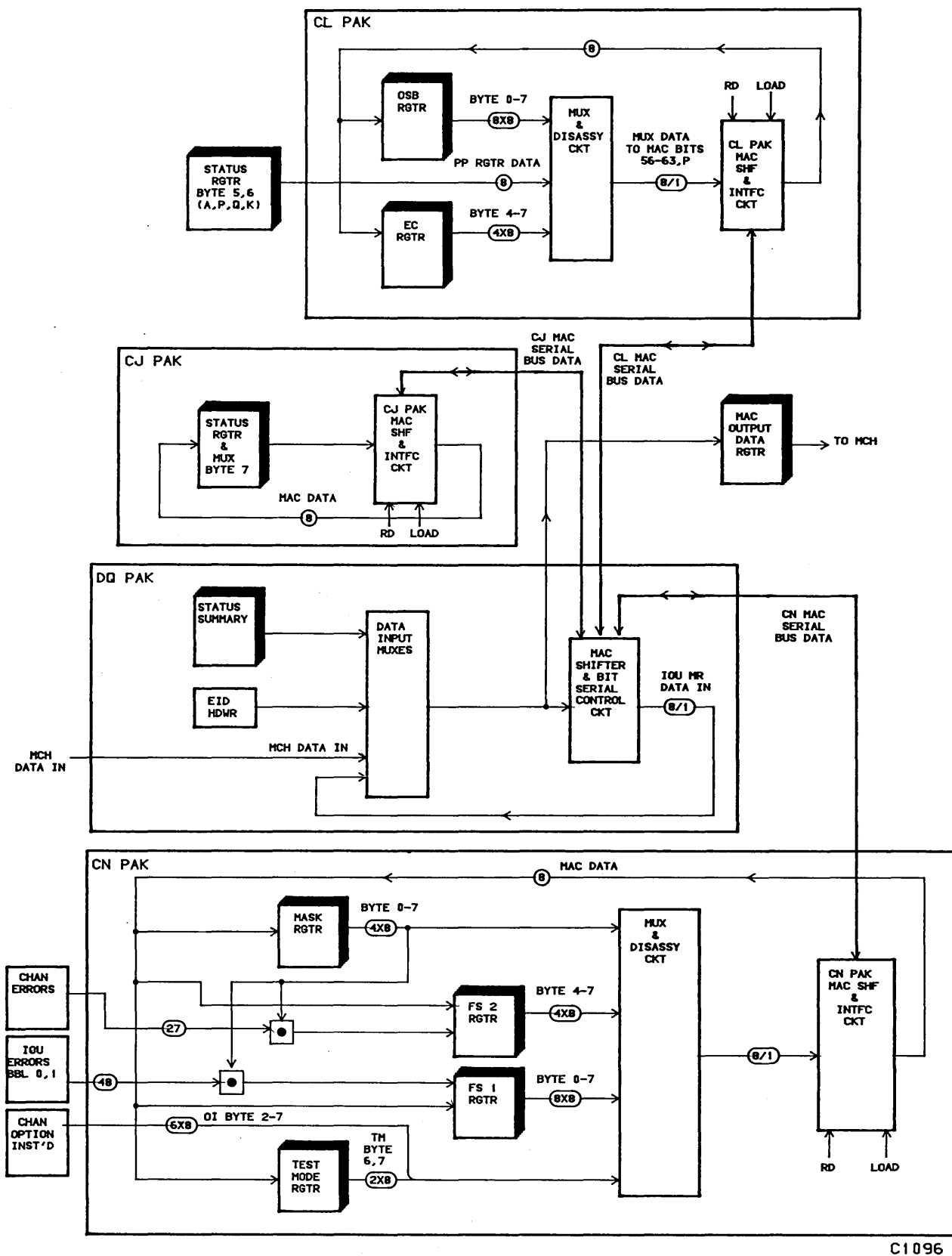
The IOU maintenance registers (MR) indicate fault status in the IOU and its environment; they also provide control functions for essential preventive and corrective maintenance.

Although most of the IOU maintenance registers indicate variety of error conditions or control functions, the fault status 1 register (FS1) indicates error in a module (pak) rather than a logical unit. Test mode register provides functions to check the parity checking hardware in IOU, whereas the fault status mask register (FSM) provides features to disable error reporting.

Ten IOU maintenance registers can be accessed via channel 17 (MCH) under the control of maintenance access control (MAC) shown in figure II-5-1. The registers are 8 bytes long (byte 0 through 7) with byte 0 being the most significant byte. When the number of bytes used is less than eight, they are right justified and zero filled for the unused bytes or bits. The only exception is the summary status byte, where all bytes are the same as shown in figure II-5-2. The addresses, MLBD references, and descriptions of the ten registers are shown in table II-5-1.

TABLE II-5-1. IOU MAINTENANCE REGISTERS

Type Code	Register Hex	Octal	No. of Bytes	MCH Access Type	Register Name	MLBD
0	00	000	1	R	Summary status byte	MAC 2.0
0	10	020	4	R	Element ID	MAC 2.0
0	12	022	6	R	Options installed	MR 2.4
0	18	030	8	R/W	Fault status mask	MR 2.4
0	21	041	8	R/W	OS (operating system) bounds	MR 2.10
0	30	060	4	R/W	Environment control	MR 2.10
0	40	100	4	R	Status	BAS 2.6, 2.12, MR 2.10, SCH 2.5
0	80	200	8	R/W	Fault status 1	MR 2.7
0	81	201	4	R/W	Fault status 2	MR 2.7
0	A0	240	2	R/W	Test mode	MR 2.5



C1096

Figure II-5-1. IOU Maintenance Registers

For a connect code and type code of zero, functions can be performed on the IOU maintenance registers according to the IOU operation code sent from MCH as follows:

IOU Operation Code

<u>Hex</u>	<u>Octal</u>	<u>Function</u>
4	4	Read
5	5	Write
6	6	Master Clear ADU
7	7	Clear Fault Status Register
C	14	Request Summary Status Byte

IOU MAINTENANCE REGISTERS TO MAC INTERFACES

Except for the status summary register and the element ID register which reside in MAC (DQ pak) for fast access, the bytes of the IOU maintenance registers reside in different paks. Refer to table III-3-2 for contents of each byte in CJ, CL, and CN paks. Access to these MR bytes are by bit serial interface with each pak containing the register bytes. Therefore, interfaces between IOU maintenance registers and MAC consist of three individual bit serial interfaces.

Each interface consists of the following three signals:

<u>Signal</u>	<u>Direction</u>	<u>MLBD Reference</u>
Read/Write	MAC to MR	MAC 2.6
Load	MAC to MR	MAC 2.6
Serial Bus Data	Bidirectional	MAC 2.6, CHL 2.0, SCH 2.5, MR 2.4, MR 2.11

For the function of master clear ADU and clear fault status register, MAC sends two signals, IOU Clear Errors and ADU Master Clear (MAC 2.1) to reset the designated registers. Refer to section III MAH for more details on bit serial control.

MAINTENANCE REGISTER FUNCTIONS

Read

The decoded read function word from MCH sets MAC into a read operation. After MAC responds with an inactive signal, the MCH is activated by the PP and sent the register address as the second control word (CW) to specify the accessing MR (first CW is ignored). MCH is deactivated by the PP after MAC accepts the second CW, and is then reactivated for an input sequence. MAC reads the request MR bytes by means of bit serial interface under the control of nanocode and then sends a Ready signal to MCH. The Ready-In signal acknowledges the

MCH input sequence. MCH responds with an Active signal to MAC. Upon receiving the Active signal, MAC sends the first data byte to MCH and sends subsequent data bytes on the receipt of an Empty signal. Any number of data words can be read. However, if a non-existent register is addressed, MAC goes into a dummy read sequence (invalid read operation) to read zeros.

Write

The decoded write function word from MCH sets MAC into a write operation. After MAC responds with an inactive signal, MCH is activated by the PP and is sent the register address as the second CW to specify the accessing MR (first CW is ignored). MCH is deactivated after MAC accepts the second CW, and is then reactivated by the PP for an output sequence. MCH sends the first data byte from the PP along with a Ready-Out signal to MAC and sends subsequent data bytes on the receipt of a Ready-In signal from MAC. MAC sends data bytes to the specified MR under the control of nanocode by using the bit serial interfaces. Any number of bytes can be sent. However, if a non-existent or a read only register is selected, MAC goes into a dummy write sequence (invalid write operation) to omit the data.

Master Clear Assembly Disassembly Unit (ADU)

Function word specified for master clear ADU function sets MAC into an immediate operation (no nanocode assistance). MAC sends a 100-ns pulse to master clear the ADU and the R register (MAC 2.1). All PPs in both barrels may be affected. This function is used during system initialization, prior to any CM read/writes.

The suggested sequence of operations, after IOU deadstart, is to master clear the memory element and then master clear the ADU. Because CM hardware does not detect IOU deadstarts, this sequence ensures that any responses to read requests made prior to IOU deadstart are cleared.

Clear Fault Status Register

Function word specified for clear status register function sets MAC into an immediate operation (no nanocode assistance). MAC sends a 100-ns IOU Clear Errors pulse (MAC 2.1) to clear fault status 1 and fault status 2 registers. Providing no other system elements are in error, summary status register bits 59 through 61 are also cleared.

Request Summary Status Byte

Function word specified for request summary status byte function sets MAC into a nanocode sequence to send the summary status byte to the MCH. The type code (TCD) and control word (CW) are not used as a fast way to read this byte. A block input should not be attempted after this function; otherwise, the MCH will timeout after the first byte is read.

Notice that this register can also be accessed by the Read function. If a read function is performed on this register (using TCD and CW), each successive word read by a block read instruction contains the same data.

MAINTENANCE REGISTER BIT DESCRIPTION

Figure II-5-2 shows the overall IOU maintenance registers and their bit assignment. Detailed bit descriptions are given in tables II-5-2 through II-5-11 and in the following paragraphs. Bits that are not available (N/A) are always zeros. Bits not used (N/U) are not assigned but may be read or written.

Summary Status Byte Register (Read only)

Bit 59 summary status, when set, indicates an error condition in any of the mainframe elements except the physical environment monitor (MAC 2.8). This bit remains set unless the error condition which was set in the unit reporting the error is cleared.

Bit 60 processor halt, when set, indicates any of the PP error bits (bits in bytes 0 through 3 of fault status 1 register) and the enable error stop bit (bit 63 of environment control register) is set (MR 2.8). This bit indicates that the PP with the error condition will halt at the end of the instruction rather than meaning the PP has halted.

Bit 61 uncorrected error sets when any error condition is detected in the fault status 1 and fault status 2 registers (MR 2.7). An uncorrected error is reported if any of the following bits set:

- Failing PP number (bytes 0-3, FS1)
- 12/16 conversion error (bit 37, FS1)
- Channel error (FS2)
- CM tag-out error (bit 52 FS1)
- OS boundary address parity error (bit 46, FS1)

Bit 61 can be cleared by either issuing a clear fault status registers function or by writing all zeros in fault status 1 and 2.

TABLE II-5-2. SUMMARY STATUS BYTE REGISTER

Byte	Bit No.	Bit Description
7	56-58	N/A
	59	Summary status
	60	Processor halt
	61	Uncorrected error
	62	N/A
	63	Physical environment monitor

IOU MAINTENANCE REGISTER

BIT	(R/W)	(R/W)	(R/W)	(RD)	(R/W)	BIT	BYTE
	TEST MODE (A0) (240) 16 8	FAULT STATUS 2 (81) (201) 16 8	FAULT STATUS 1 (80) (200) 16 8	STATUS (40) (100) 16 8	ENVIRONMENT CONTROL (30) (060) 16 8		
00			N/A			00	0
01			N/A			01	
02			N/A			02	
03			PP 4 ERR			03	
04			PP 3 ERR			04	
05			PP 2 ERR			05	
06			PP 1 ERR			06	
07			PP 0 ERR			07	
08			N/A			08	1
09			N/A			09	
10			N/A			10	
11			PP 11 ERR			11	
12			PP 10 ERR			12	
13			PP 7 ERR			13	
14			PP 6 ERR			14	
15			PP 5 ERR			15	
16			N/A			16	2
17			N/A			17	
18			N/A			18	
19			PP 24 ERR			19	
20			PP 23 ERR			20	
21			PP 22 ERR			21	
22			PP 21 ERR			22	
23			PP 20 ERR			23	
24			N/A			24	3
25			N/A	N/A		25	
26			N/A			26	
27			PP 31 ERR			27	
28			PP 30 ERR			28	
29			PP 27 ERR			29	
30			PP 26 ERR			30	
31			PP 25 ERR	BATTERY ON-LINE		31	
32		CHAN 7 ERROR	ERROR CL	N/A	N/A	32	4
33		6	ERROR CR		N/U	33	
34		5	FIRMWARE ERROR		N/U	34	
35		4	ERROR CM		PP NUMBER	35	
36		3	ERROR CP			36	
37		2	CVRSN ERROR		PP NUMBER	37	
38		1	N/U	PP RGTR A,P,Q,K		38	
39		CHAN 0 ERROR	PPM DATA-IN ERR	PP RGTR A,P,Q,K		39	
40		CHAN 17 ERROR	N/U	PP RGTR A,P,Q,K	N/A	40	5
41		N/A			N/U	41	
42		CHAN 15 ERROR			N/U	42	
43		N/A			CHAN NUMBER	43	
44		CHAN 13 ERROR	N/U			44	
45		12	OSB VIOLATION		CHAN NUMBER	45	
46		11	OSB ADPS PE			46	
47		CHAN 10 ERROR	N/U			47	
48	N/A	CHAN 27 ERROR	CM DATA OUT ERR		N/U	48	6
49	N/A	26	UNCOR CM RD ERR		N/U	49	
50	N/U	25	UNCOR CM WR ERR		N/U	50	
51	LOGICAL BBL SEL	24	CH REJECT		LOAD MODE	51	
52	LOGICAL PP NUM	23	CH TAG OUT ERR		DUMP MODE	52	
53		22	CH RESP CODE ERR		IDLE MODE	53	
54		21	N/U		RGTR SEL 0=P	54	
55	LOGICAL PP NUM	CHAN 20 ERROR	N/U	PP RGTR A,P,Q,K	1=Q 2=K 3=A	55	
56	N/A	N/A	N/U	LDS BIT	BBL 1 BLK	56	7
57	N/A	N/A		N/A	BBL 0 BLK	57	
58	N/U	N/U		BBL RECONFIG	EN DS/DUMP/IDLE	58	
59		RI 2/3		PP RECONFIG	ENBL TEST MODE	59	
60		CHAN 33 ERROR			ENBL OSB CHECK	60	
61		32			ENBL R+A TO PPM	61	
62		31			N/U	62	
63		CHAN 30 ERROR	N/U	PP RECONFIG	ENBL ERR STOP	63	

NOTES:
 N/A - BITS NOT AVAILABLE WILL ALWAYS BE ZEROES
 N/U - THESE BITS ARE NOT ASSIGNED BUT MAY BE READ OR WRITTEN
 (RD) - READ ONLY REGISTER
 (R/W) - READ/WRITE REGISTER
 UNANNOTATED BYTES = N/A

C1089

IOU TEST CODES

- 00 N/U
- 01 CHAN TO PP (GEN) CP
- 02 PP TO CHAN (GEN) CP
- 03 PPM TO R RGTR (GEN) CR
- 04 PPM (CHECK) CR
- 05 mCODE DATA (CHECK) CP
- 06 PPM (GEN) CM
- 07 CM FCIN (GEN) CP

- 08 Y RGTR (GEN) CM
- 09 A RGTR (GEN) CR
- 0A SHF ROM (CHECK) CR
- 0B Q RGTR (GEN) CR
- 0C P RGTR (GEN) CR
- 0D G MUX (GEN) CR
- 0E R TO Y (GEN) CL
- 0F N/U

- 10 N/U
- 11 ADPS TO CMC (*) CL
- 12 N/U
- 13 DATA TO CMC (*) CM
- 14 OSB ADPS (CHECK) CL
- 15 TAG IN (CHECK) DC
- 16 RESP CODE (CHECK) CN
- 17 CHAN 15 DATA BUS (CHECK) CK

(*) 'HI' FORCED FOR PAR BIT AT GENERATOR
 (GEN) PARITY INVERTED ON GENERATOR
 (CHECK) PARITY INVERTED ON CHECKING

Figure II-5-2. IOU Maintenance Register Bit Assignment (1 of 2)

IOU MAINTENANCE REGISTER

		(R/W)		(R/W)		(RD)		(RD)		(RD)			
BYTE	BIT	OS BOUNDS		FAULT STATUS MASK		OPTIONS INSTALLED		ELEMENT ID		STATUS SUMMARY		BIT	BYTE
		(21) 16	(041) 8	(18) 16	(030) 8	(12) 16	(022) 8	(10) 16	(020) 8	(00) 16	(000) 8		
0	00	N/A		N/A		N/A				SAME AS BYTE 7		00	0
	01	N/A		N/A								01	
	02	N/A		N/A								02	
	03	PP 4 BV		PP 4 MV								03	
	04	PP 3 BV		PP 3 MV								04	
	05	PP 2 BV		PP 2 MV								05	
	06	PP 1 BV		PP 1 MV								06	
	07	PP 0 BV		PP 0 MV								07	
1	08	N/A		N/A						SAME AS BYTE 7		08	1
	09	N/A		N/A								09	
	10	N/A		N/A								10	
	11	PP 11 BV		PP 11 MV								11	
	12	PP 10 BV		PP 10 MV								12	
	13	PP 7 BV		PP 7 MV								13	
	14	PP 6 BV		PP 6 MV								14	
	15	PP 5 BV		PP 5 MV								15	
2	16	N/A		N/A		N/A				SAME AS BYTE 7		16	2
	17	N/A		N/A		N/A						17	
	18	N/A		N/A		N/A						18	
	19	PP 24 BV		PP 24 MV								19	
	20	PP 23 BV		PP 23 MV		PP 25-31						20	
	21	PP 22 BV		PP 22 MV		PP 20-24						21	
	22	PP 21 BV		PP 21 MV		PP 5-11						22	
	23	PP 20 BV		PP 20 MV		PP 0-4						23	
3	24	N/A		N/A		CHAN 7				SAME AS BYTE 7		24	3
	25	N/A		N/A		6						25	
	26	N/A		N/A		5						26	
	27	PP 31 BV		PP 31 MV		4						27	
	28	PP 30 BV		PP 30 MV		3						28	
	29	PP 27 BV		PP 27 MV		2						29	
	30	PP 26 BV		PP 26 MV		1						30	
	31	PP 25 BV		PP 25 MV		CHAN 0						31	
4	32			CHAN 7 HV		CHAN 17				SAME AS BYTE 7		32	4
	33			6 HV		6						33	
	34			5 HV		CHAN 15						34	
	35			4 HV		N/A						35	
	36			3 HV		CHAN 13						36	
	37			2 HV		12						37	
	38			1 HV		11						38	
	39			CHAN 0 HV		CHAN 10						39	
5	40	N/A		CHAN 17 HV		CHAN 27				SAME AS BYTE 7		40	5
	41			N/A		26						41	
	42			CHAN 15 HV		25						42	
	43			N/A		24						43	
	44			CHAN 13 HV		23						44	
	45			12 HV		22						45	
	46	OSB ADRS		11 HV		21						46	
	47	OSB ADRS		CHAN 10 HV		CHAN 20						47	
6	48	OSB ADRS		CHAN 27 HV		N/A				SAME AS BYTE 7		48	6
	49			26 HV		N/A						49	
	50			25 HV		N/A						50	
	51			24 HV		N/A						51	
	52			23 HV		CHAN 33						52	
	53			22 HV		32						53	
	54	OSB ADRS		21 HV		31						54	
	55	OSB ADRS		CHAN 20 HV		CHAN 30						55	
7	56	OSB ADRS		N/A		BATTERY BACKUP				N/A		56	7
	57			N/A		N/A						57	
	58			N/A		N/A						58	
	59			RI 2/3 HV		N/A						59	
	60			CHAN 33 HV		RI 2/3						60	
	61			32 HV		RI 0/1						61	
	62			31 HV		2 PORT MUX						62	
	63	OSB ADRS		CHAN 30 HV		CCS45 CONTROLLER						63	

NOTES
 N/A - BITS NOT AVAILABLE WILL ALWAYS BE ZEROES
 N/U - THESE BITS ARE NOT ASSIGNED BUT MAY BE READ OR WRITTEN
 (RD) - READ ONLY REGISTER
 (R/W) - READ/WRITE REGISTER
 MV - MASK VECTOR
 BV - BIT VECTOR
 UNANNOTATED BYTES = N/A

C1088

Figure II-5-2. IOU Maintenance Register Bit Assignment (2 of 2)

Bit 63 physical environment monitor, when set, indicates a problem condition is reported by an external dew point recorder (if installed).

Element ID Register (Read Only)

Element ID bits are hardwired in MAC element ID select mux (MAC 2.8) to indicate equipment type, model number, and individual serial number as shown in table II-5-3.

TABLE II-5-3. ELEMENT ID REGISTER

Byte	Bit No.	Bit Description	Byte	Bit 0	Bit Description
0-3	0-31	N/A	5	40-47	Model no. 13 ₁₆ =830 14 ₁₆ =810
4	32-39	Equipment type No 02 ₁₆ for IOU	6-7	48-63	Serial no. (in packed decimal)

Options Installed Register (Read Only)

A set bit in this register indicates that the option shown in table II-5-4 is installed.

Fault Status Mask Register (Read/Write)

A set bit in the fault status mask register (FSM) indicates error in the corresponding fault status registers bit has been masked to disable error reporting. This feature of disable error reporting for a given PP or channel enables the IOU to continue running in degraded mode (without using that failing hardware). This is made possible by masking all error reporting from that PP or channel by setting the appropriate mask vector bit in the FSM shown in table II-5-5.

PP errors are masked before the error is decoded to a failure PP number (MR 2.6). Therefore, when a given PP's Mask Vector is set, the failing PP number and the failing pak or error type(s) associated with that PP are masked in the fault status registers. However, of the errors that are decoded from the PP, only the PP pak errors (byte 4, FS1) and OS bounds violation (bit 45, FS1) can be masked; the ADU errors (byte 6 and 7 of FS1) cannot be masked. Since ADU hardware is shared across PPs, error masking to a PP is not meaningful. However, a PP that has been found in error can be prevented by software from performing any CM access. Thus that PP would not log any ADU errors.

TABLE II-5-4. OPTIONS INSTALLED REGISTER

Byte	Bit No.	Bit Description	Byte	Bit No.	Bit Description
0-1	0-15	N/A		38	Channel 11
				39	Channel 10
2	16-19	N/A			
	20	PP25-315	5	40	Channel 27
	21	PP20-24		41	Channel 26
	22	PP5-11		42	Channel 25
	23	PP0-4		43	Channel 24
				44	Channel 23
3	24	Channel 7		45	Channel 22
	25	Channel 6		46	Channel 21
	26	Channel 5		47	Channel 20
	27	Channel 4			
	28	Channel 3	6	48-51	N/A
	29	Channel 2		52	Channel 33
	30	Channel 1		53	Channel 32
	31	Channel 0		54	Channel 31
				55	Channel 30
4	32	Channel 17		56	Battery backup
	33	N/A	7	57-59	N/A
	34	Channel 15		60	Radial interface 2/3
	35	N/A		61	Radial interface 0/1
	36	Channel 13		62	Two port multiplexer
	37	Channel 12		63	CC545 controller

Operating System Bounds Register (Read/Write)

The operating system bounds register (OSB) allows the division of CM address space into upper and lower regions for dual state operation. When any PP is executing CM write or exchange operations, its CM address (bits 36 through 63) is compared to the OS boundary address shown in figure II-5-3 and checked with the appropriate PP bit vector for validity (BAS 2.3). If the address specified is outside that PP's allowed region and the enable OS bounds checking bit (bit 60 of the EC register, MR 2.10) is set, then bit 45 (OS bounds violation) of fault status 1 sets, along with the corresponding failing PP number. Furthermore, if the enable error stop bit (bit 63 of EC register, MR 2.10) is set, the failing PP halts at the end of the CM instruction.

Each PP bit vector indicates which portion of memory that PP may address. A set bit indicates the PP CM address is less than OS boundary as shown in table II-5-6. Therefore, that PP must access the lower region. A cleared bit indicates PP CM address is greater than or equal to OS boundary. Hence, that PP must access the upper region. PPs can be dynamically switched between regions by changing their bit vector without changing the boundary. The OS boundary defines the boundary between the two regions with an accuracy of 2^{10} (1024) words or 2^{13} (8192) bytes.

TABLE II-5-5. FAULT STATUS MASK REGISTER

Byte	Bit No.	Bit Description	Byte	Bit No.	Bit Description
0	0-1	N/A		34	Channel 5 mask vector
	2	N/U		35	Channel 4 mask vector
	3	Barrel 0 PP4 mask vector		36	Channel 3 mask vector
	4	Barrel 0 PP3 mask vector		37	Channel 2 mask vector
	5	Barrel 0 PP2 mask vector		38	Channel 1 mask vector
	6	Barrel 0 PP1 mask vector		39	Channel 0 mask vector
	7	Barrel 0 PP0 mask vector		40	Channel 17 mask vector
1	8-9	N/A	5	41	N/U
	10	N/U		42	Channel 15 mask vector
	11	Barrel 0 PP11 mask vector		43	N/U
	12	Barrel 0 PP10 mask vector		44	Channel 13 mask vector
	13	Barrel 0 PP7 mask vector		45	Channel 12 mask vector
	14	Barrel 0 PP6 mask vector		46	Channel 11 mask vector
	15	Barrel 0 PP5 mask vector		47	Channel 10 mask vector
2	16-17	N/A	6	48	Channel 27 mask vector
	18	N/U		49	Channel 26 mask vector
	19	Barrel 1 PP4 mask vector		50	Channel 25 mask vector
	20	Barrel 1 PP3 mask vector		51	Channel 24 mask vector
	21	Barrel 1 PP2 mask vector		52	Channel 23 mask vector
	22	Barrel 1 PP1 mask vector		53	Channel 22 mask vector
	23	Barrel 1 PP0 mask vector		54	Channel 21 mask vector
3	24-25	N/A	7	55	Channel 20 mask vector
	26	N/U		56	N/U
	27	Barrel 1 PP11 mask vector		57	N/U
	28	Barrel 1 PP10 mask vector		58	N/U
	29	Barrel 1 PP7 mask vector		59	Radial interface
	30	Barrel 1 PP6 mask vector			2/3 mask vector
	31	Barrel 1 PP5 mask vector		60	Channel 33 mask vector
4	32	Channel 7 mask vector		61	Channel 32 mask vector
	33	Channel 6 mask vector		62	Channel 31 mask vector
				63	Channel 30 mask vector
Note: The barrel numbers are physical barrels and the PP numbers are logical PPs.					

TABLE II-5-6. OPERATING SYSTEM BOUNDS REGISTER

Byte	Bit No.	Bit Description	Byte	Bit No.	Bit Description
0	0-2	N/A		20	Barrel 1 PP3 bit vector
	3	Barrel 0 PP4 bit vector		21	Barrel 1 PP2 bit vector
	4	Barrel 0 PP3 bit vector		22	Barrel 1 PP1 bit vector
	5	Barrel 0 PP2 bit vector		23	Barrel 1 PP0 bit vector
	6	Barrel 0 PP1 bit vector			
	7	Barrel 0 PP0 bit vector	3	24-26	N/A
1	8-10	N/A		27	Barrel 1 PP11 bit vector
	11	Barrel 0 PP11 bit vector		28	Barrel 1 PP10 bit vector
	12	Barrel 0 PP10 bit vector		29	Barrel 1 PP7 bit vector
	13	Barrel 0 PP7 bit vector		30	Barrel 1 PP6 bit vector
	14	Barrel 0 PP6 bit vector		31	Barrel 1 PP5 bit vector
	15	Barrel 0 PP5 bit vector	4	32-39	N/A
2	16-18	N/A	5-7	40-45	N/U
	19	Barrel 1 PP4 bit vector		46-63	OS boundary address

Note: The barrel numbers are physical barrels and the PP numbers are logical PPs.

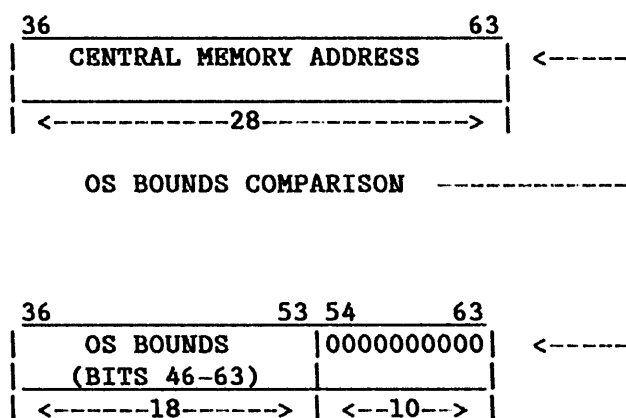


Figure II-5-3. Operating System Bounds Comparison

Only the following IOU instructions have their CM address verified:

1000	Read and set lock
1001	Read and clear lock
0026	Exchange jump (FWA only is verified)
0062	Central write direct
1062	Central write direct long
0063	Central write memory
1063	Central write memory long

If a bounds violation occurs during an exchange instruction (26xx), the Exchange Request signal to the processor is blocked; for the remaining instructions listed above, the write request to CM is blocked by a Block Write signal (MEM 2.5). Otherwise, all the above instructions execute normally. However, if bit 63 of EC register (enable error stop) is set when the instruction exists, the PP causing the violation hangs when it attempts to read the next instruction. The K register indicates the CM instruction that caused the violation.

The exchange jump and monitor exchange jump (2600 and 2610) instructions have only the FWA (first word address) of their exchange package compared with the OS boundary. The monitor exchange jump to MA (2620) instruction is excluded from bounds checking. Also, an exchange package could extend across the boundary.

Before any CM write or exchange operations are attempted, the OS bounds register must be initialized and set up by software. Care must be taken when changing the OS boundary, because only one byte is updated every trip of the MCH write instruction. While the boundary is in an intermediate stage (new byte 6 and old byte 7), all PPs must be held from writing or exchanging.

Environment Control Register (Read/Write)

Environment control register (EC) specifies which channel PP, and internal register (A, P, Q, K) are to be latched into the status register. Modes of operation and enable signals to check various hardware are also latched in the EC register as shown in table II-5-7.

Bits 35 to 39 PP Number define the logical PP number for load, dump, or idle PP operation. They also define the PP number with the selected register (A, P, Q, K) which will be latched into the status register.

If illegal PP number (12 to 17 or 32 to 37) is selected, no data latches into the status register or no PP is load/dump or idle deadstarted. However, if the PP number selected is not installed, all ones are latched into the status register.

Bits 43 to 47 Channel Number specify the channel number for load or dump PP functions.

TABLE II-5-7. ENVIRONMENT CONTROL REGISTER

Byte	Bit No.	Bit Description	Byte	Bit No.	Bit Description
0-3	0-31	N/A		51	Load mode
				52*	Dump mode
4	32	N/A		53	Idle mode
	33*	N/U		54-55	Register select (A,P,Q,K)
	34	N/U			
	35-39	PP number	7	56	+BBL 1 block mode select
				57	+BBL 0 block mode select
5	40-41	N/A		58*	Enable deadstart/dump/idle
	42	N/U		59*	Enable test mode
	43-47	Channel number		60*	Enable OS bounds checking
				61*	Enable (R)+(A) to PP memory
6	48-49*	N/U		62	N/U
	50	N/U		63*	Enable error stop

* Bit cleared by master clear from deadstart function.

When bit 51 Load Mode and bit 58 of the EC register (enable load/dump/idle) are set, the PP number specified in bits 35 through 39 above is forced into a block input instruction (0071) over the channel specified in bits 43 through 47.

When bit 52 Dump Mode and bit 58 of the EC register are set, the PP number specified is forced into a block output instruction (0073) over the channel specified.

NOTE

If bits 51, 52, and 58 are all set,
the PP is forced into dump mode.

When bit 53 Idle Mode and bit 58 of the EC register are set, the specified PP is forced into idle mode. When idled, a PP can neither perform PPM instructions nor communicate with the channels. The only way to regain control of an idled PP is through master clear, load, or dump deadstart.

A PP only goes into idle mode at the end of the instruction in which it receives the Idle signal (BAS 2.11). The PP waits until the instruction is finished to ensure that CM control are not left in an undesired state. When PP is in idle mode, the K register for that PP is set to all ones (that is 1077). If load/dump mode operation is performed over an inactive channel, the PP forced into load/dump mode exits right away.

NOTE

If bit 51 and/or bit 52 and both bits 53 and 58 are set, the PP goes into idle mode after exiting from its block input or block output instruction.

Bit 54 and 55 Register Select - These two bits specify register to be latched into status register. The two bits are decoded as shown below:

Bits		<u>Register Selected</u>
<u>54</u>	<u>55</u>	
0	0	P register
0	1	Q register
1	0	K register
1	1	A register

Bits 56 and 57 are used in controlling the ADU priority. Bit 56 controls physical barrel 1. Bit 57 controls physical barrel 0. Master clear sets both bits to zero to select BLOCK mode. For further details, refer to ADU Priority in Part IV Section 1.

Bit 58 Enable Load/Dump/Idle enables the load, dump or idle mode, and should be set after the mode bits (51 to 53) are set. It is cleared automatically by the hardware (DST 2.11) within 600 ns after it is set. If this bit is set and no mode bits are set, nothing happens and the Enable bit is cleared by master clear.

Bit 59 Enable Test Mode enables test signal from the test mode register to check the parity checking hardware in the IOU.

Bit 60 Enable OS Bounds Checking enables OS bounds violations to report in bit 45 of FS1. If bit 60 is set and a violation has occurred, write or exchange requests to CM and the CPU are blocked.

Bit 61 Enable (R)+(A) to PPM allows testing of R+A adder without using central memory. When this bit is set, the 0025 instruction stores the result of R+A adder instead of the R register. Bits 36 through 57 of R+A adder which are enabled by select bit 61 in the RA mux, are stored in PP memory in the same format as a normal 0025 instruction (BAS 2.3).

Bit 63 Enable Error Stop enables the halt of a PP on completion of an instruction when an error bit for that PP is set in the FS1.

Status Register (Read Only)

Status register specifies the status of the IOU as shown in table II-5-8.

TABLE II-5-8. STATUS REGISTER

Byte	Bit No.	Bit Description
0-3	0-31	N/A
4	32-37	N/A
5-6	38-55	Internal register (A,P,Q,K)
7*	56	LDS bit
	57	N/A
	58	N/A
	59	Barrel reconfiguration
	60-63	PP reconfiguration
* Byte latched by master clear.		

Bit 38 to 55 Internal Register latches contents of the selected register (A, P, Q, K register data from the MAC data register and a selected PP (BAS 2.6, 2.12)).

Bit 56 LDS Bit indicates, when set, that a long deadstart sequence was used to deadstart IOU (SCH 2.5).

Bit 59 to 63 Reconfiguration Bits indicate PP and barrel of IOU used at deadstart time (SCH 2.5).

Bit 59 indicates which physical barrel is defined as logical barrel 0. Bits 60 through 63 specify physical PP memory that is designated as logical memory 0 (PPM 2.0).

If an illegal PP or a not-installed reconfiguration is selected, hardware assumes a value of zero for reconfiguration. Zero is also stored in status register.

Fault Status 1 Register (Read/Write)

With the exception of the errors listed below, all error conditions in the IOU are unconditionally decoded into a failing PP number:

- 12/16 Conversion error (bit 37, FS1)
- Channel error (FS2)

- CM Tag-Out error (bit 52 FS1)
- OS Boundary Address parity error (bit 46, FS1)

At the same time, another bit is also set to indicate the type of error or the module on which the error was detected as shown in table II-5-9.

TABLE II-5-9. FAULT STATUS 1 REGISTER*

Byte	Bit No.	Bit Description	Byte	Bit No.	Bit Description
0	0-2	N/A	4	32	CL pak error
	3	Barrel 0 PP4 error		33	CR pak error
	4	Barrel 0 PP3 error		34	Firmware error (microcode)
	5	Barrel 0 PP2 error		35	CM pak error
	6	Barrel 0 PP1 error		36	CP pak error
	7	Barrel 0 PP0 error		37	12/16 Conversion error
				38	N/U
1	8-10	N/A	5	39	PP memory data-in error
	11	Barrel 0 PP11 error		40-44	N/U
	12	Barrel 0 PP10 error		45	OS bounds violation
	13	Barrel 0 PP7 error		46	OS boundary address parity error
	14	Barrel 0 PP6 error		47	N/U
	15	Barrel 0 PP5 error			
2	16-18	N/A	6	48	CM (ADU) data out error
	19	Barrel 1 PP4 error		49	Uncorrected CM read error
	20	Barrel 1 PP3 error		50	Uncorrected CM write error
	21	Barrel 1 PP2 error		51	CM reject
	22	Barrel 1 PP1 error		52	CM tag out error
	23	Barrel 1 PP0 error		53	CM response code error
3	24-26	N/A	7	54	N/U
	27	Barrel 1 PP11 error		55	N/U
	28	Barrel 1 PP10 error			
	29	Barrel 1 PP7 error		56-63	N/U
	30	Barrel 1 PP6 error			
	31	Barrel 1 PP5 error			

* This register cleared by long deadstart sequence (LDS).

Bit 0 to 31 PP Error. Each bit, when set, indicates which PP detected the error.

Bit 32 CL Pak Error, when set, indicates a parity error is detected on transfers from R register to Y mux or from R+A register to Y mux (BAS 2.3, MR 2.10).

Bit 33 CR Pak Error, when set, indicates a parity error is detected (BAS 2.12). Possible locations of this error are:

- A register shift control ROM data (BAS 2.1)
- PP memory address bits (BAS 2.5)
- PP memory data out (BAS 2.2)
- A register barrel data at rank 8 (BAS 2.0)
- R register barrel data at rank 8 (BAS 2.2)
- P register barrel data at rank 8 (BAS 2.5)
- Q register barrel data at rank 8 (BAS 2.4)

Bit 34 Firmware (microcode) Error, when set, indicates a parity error detected in the control (microcode) firmware (BAS 2.11). Parity checking is performed on all bits of the micrand field (BAS 2.8). Firmware data bits 00 through 48, 59, 60, 71 to 87 are checked for correct parity at slot time (rank 9). Firmware address bits 49 through 58 (BR1 field) and bits 61 to 70 (BR2 field) are checked one major cycle later at slot time. The selected Branch Address (BR1 or BR2) is also checked for parity before addressing the ROMs. These conditions cause the failing PP to be unconditionally halted on the following trip. This bit also sets if the instruction from memory has a parity error. As a result, the PP is unconditionally halted.

Bit 35 CM Pak Error, when set, indicates parity error is detected on either PP memory write data in or PP memory read data out at rank 8 (PPM 2.1).

Bit 36 CP Pak Error, when set, indicates a parity error is detected in either the firmware (bit 34 set) or channel to Y mux data (BAS 2.11).

Bit 37 Conversion Error, when set, indicates an conversion error is detected on channel data at channel input conversion macrocell or channel output conversion macrocell (BAS 2.10) on the CP pak. This error is not associated with a PP. That is, no PP error bit sets when this bit is set).

Bit 39 PP Memory Data-In Parity Error, when set, indicates a parity error in the data being written into the PP memory pak. This bit also sets bit 35, CM pak Error, and bit 48, ADU Data Out Error (PPM 2.2).

Bit 45 OS Bounds Violation, when set, indicates a PP has attempted to write or exchange at a CM address outside the region allowed by the OS bound register. This bit sets only if bit 60 of the EC register (enable OSB) is set (MR 2.10).

Bit 46 OS Boundary Address Parity Error, when set, indicates a parity error is detected on OS boundary address output from OS bound register (MR 2.10). This address is the boundary for all PPs in both barrels.

Bit 48 CM (ADU) Data Out Error, when set, indicates a parity error is detected on data coming from the ADU (CM data) into the PP memory (PPM 2.2).

Bit 49 Uncorrected CM Read Error is decoded from CM response code 0 to 2 in the response code decoder when a response to IOU signal is being received (MR 2.6). The memory element reports this status (set response code=5) if it detects:

- Multiple SECDED (DBE) error or a read parity error.
- Any parity error (except on function code or tag in signal).

Bit 50 Uncorrected CM Write Error is decoded from CM response code 0 to 2 in the response code decoder when a response to IOU signal is being received (MR 2.6). The memory element reports this status (set response code=1) on a write function if it detects (MR 2.13):

- Multiple SECDED (DBE) error
- Any parity error (except on function code or tag in signal)
- Bounds fault

Bit 51 CM Reject is decoded from CM response code 0 to 2 in the response code decoder when a response to IOU signal is being received (MR 2.6). If a reject is received on a CM read request, ADU blocks response for that request; PP that made the request hangs. The memory element reports this status (set response code=7) if it detects the following:

- Function code parity error
- Illegal function code
- Data-in error on an interrupt

Bit 52 CM Tag Out Error, when set, indicates ADU hardware has detected a parity error in tag bits from CM CT macrocell (MR 2.7). This error condition cannot be decoded into a failing PP number because tags define the number of PPs performing CM operation. For a read function, these tag bits determine into which buffer the CM data is to be written. If a tag error is sensed, ADU blocks response for that request; the PP that made the request hangs.

Bit 53 CM Response Code Error, when set, indicates ADU hardware has detected a parity error in response code from CM CT macrocell (MR 2.6). If the error condition is sensed by ADU on a read request, ADU blocks response for that request forcing the requesting PP to hang.

Fault Status 2 Register (Read/Write)

This register reports parity errors detected on the channels and on radial interfaces 2 and 3 (MR 2.7). Each channel has its own bit in the register as shown in table II-5-10.

TABLE II-5-10. FAULT STATUS 2 REGISTER*

Byte	Bit No.	Bit Description	Byte	Bit No.	Bit Description
0-3	0-31	N/A	6	48	Channel 27 error
				49	Channel 26 error
4	32	Channel 7 error		50	Channel 25 error
	33	Channel 6 error		51	Channel 24 error
	34	Channel 5 error		52	Channel 23 error
	35	Channel 4 error		53	Channel 22 error
	36	Channel 3 error		54	Channel 21 error
	37	Channel 2 error		55	Channel 20 error
	38	Channel 1 error			
	39	Channel 0 error	7	56	N/A
				57	N/U
5	40	Channel 17 error		58	N/U
	41	N/A		59	Radial interface 2/3
	42	Channel 15 error		60	Channel 33 error
	43	N/A		61	Channel 32 error
	44	Channel 13 error		62	Channel 31 error
	45	Channel 12 error		63	Channel 30 error
	46	Channel 11 error			
	47	Channel 10 error			
* This register cleared by long deadstart sequence (LDS).					

Channel parity is checked whenever the channel is full. If any channel parity error is detected, a one shot signal from the error reporting register (CHL 2.2, SCH 2.0, DSC 2.1) is sent to the fault status 2 register (FS2). The FS2 or error reporting register can be cleared independently of one another. The error reporting register, even though it can be cleared only when the channel is not full, does not have to be cleared to allow reporting of new error to the FS2. The FS2 can be cleared anytime by writing or masking. Data in the channel registers can come from the PP or the external device. Checking is disabled if the parity switch for that channel is off.

Test Mode Register (Read/Write)

The test mode register defined in table II-5-11, checks the parity hardware in IOU. This register allows the inversion of parity checker results. The parity bits involved are generally not altered. Thus, bad parity does not pass through the hardware, and generally only one error bit sets in response to one of the test codes shown in table II-5-12. Test code bit is generated from the test mode decoder when bit 59 of EC register (enable test mode) is set (MR 2.9). However, the register may be read and written even if bit 59 of EC register is not set.

TABLE II-5-11. TEST MODE REGISTER

Byte	Bit No.	Bit Description
0-5	0-47	N/A
6	48-49	N/A
	50	N/U
	51	Logical barrel
	52-55	Logical PP
7	56-57	N/A
	58	N/U
	59-63	Test code

Bit 51 Barrel Select indicates logical barrel to be tested.

Bit 52 to 55 PP Select selects the logical PP (00 to 11) to be tested in barrel indicated by bit 51.

Bit 59 to 63 Test Mode code are decoded as shown in table II-5-12 to provide functions for testing error detection logic in IOU; Test mode bit is active when bit 59 of EC register is set and a legal PP number is set in bits 52 through 55.

TABLE II-5-12. TEST MODE FUNCTION

Test Code Bits 60 - 63	Function	MLBD Reference
00	Not used	
01	Invert channel to PP parity	BAS 2.10
02	Invert PP to channel parity	BAS 2.9
03	Invert PPM to R register parity	BAS 2.1
04	Invert PPM parity	BAS 2.1
05	Invert microcode data parity	PPM 2.8
06	Invert PPM parity rank 9	MEM 2.1
07	Invert CM function code parity	PPM 2.5
10	Invert Y register parity	BAS 2.2
11	Invert A register parity	BAS 2.0
12	Invert shift control ROM parity	BAS 2.1
13	Invert Q register parity	BAS 2.4
14	Invert P register parity	BAS 2.5
15	Invert G mux parity	BAS 2.5
16	Invert R to Y parity	BAS 2.3
17	Not used	
20	Not used	
21	Force zero on CM address parity	BAS 2.3
22	Not used	
23	Force zero on CM data in parity	ADU 2.1
24	Invert OS bounds address parity	MR 2.10
25	Invert tag in parity	MEM 2.3
26	Invert response code parity	MR 2.6
27	Invert channel 15 data bus parity	DST 2.10

SECTION II-6

TWO PORT MULTIPLEXER

=====

The IOU contains a two port multiplexer (TPM) which provides serial communication capability between any PP and two attached terminals through the dedicated channel 15*.

The TPM consists of an IOU internal interface on channel 15 and two standard RS-232-C External Interfaces, one each on port 0 and port 1. Port 0 is reserved for use by the standard system console; port 1 is reserved for engineering services maintenance operations.

GENERAL DESCRIPTION

TPM is an asynchronous communication interface connected with channel 15. Ports 0 and 1 have an RS-232-C interface, an eight position BAUD RATE SELECTION switch (110, 300, 600, 1200, 2400, 4800, 9600, and 19200 baud), and a four position PORT OPTIONS switch mounted on the two port mux box in the back of the mainframe. Each port has a 64-character output buffer and a three-character input buffer.

The TPM supports data communication using ASCII code only. Special features supported by the TPM are:

- Auto answer
- Remote power control
- Remote deadstart
- Calendar clock

The TPM supports any combination of CC555 or equivalent terminals and modems on ports 0 and 1.

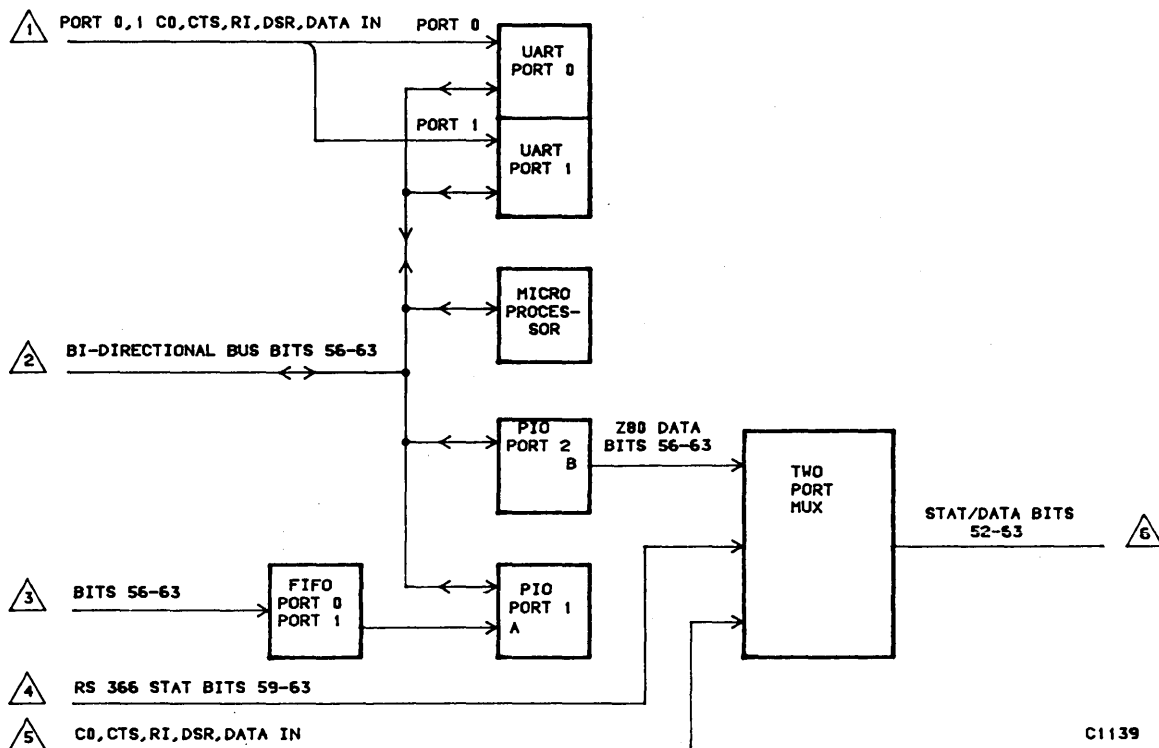
The calendar clock provides the time of day, month, and year. The PPs set or read the calendar clock through channel 15.

An RS-366 interface is on port 1 only which provides the auto-dial-out feature.

The TPM shown in figure II-6-1 contains five major components:

- Universal asynchronous receiver/transmitter (UART) (DST 2.1)
- Parallel input/output port 2 (PIO 2) (DST 2.1)

* All channel numbers are in octal.



C1139

△1, △4, & △5 From terminal

△2 To microprocessor RAMs, ROMs, and calendar clock

△3 From channel 15 or keyboard

△6 To channel 15

Figure II-6-1. Two Port Mux

- Microprocessor (DST 2.0)
- First in first out memory (FIFO) (DST 2.3)
- Parallel input/output port 1 (PIO 1) (DST 2.1)

One UART is on external port 0 (DST 2.1), the other is on external port 1. The UART converts the asynchronous serial binary characters from the terminal (RS232-C interface) to a parallel format for the PP and vice versa. The parallel data from the UART is input/output to the microprocessor bidirectional data bus.

The PIO 1 port A (DST 2.1) selects data from the FIFO and sends it to the microprocessor bidirectional data bus which feeds the UARTs (DST 2.1). The data is then sent over the RS232-C interface.

PIO 2 ports A and B (DST 2.0) and associated circuitry control the data flow from the microprocessor bidirectional data bus to the barrel 0/1 channel 15 data bus.

The microprocessor (DST 2.0) decodes function codes, sets the functions, and monitors the operation of the other major TPM components.

The FIFO (DST 2.3) is a buffer memory which stores PP data until it can be output over the slower external RS232-C interface.

EXTERNAL INTERFACE

RS232-C INTERFACE

Both ports of the TPM use EIA Standard RS232-C asynchronous transmission. All data contains a start bit, 5 to 8 data bits, 1 or 2 stop bits, and either odd/even parity or no parity. To make the terminal universal, the bits per word, parity mode, and number of stop bits are software programmable. The baud rate selection for port 1 is independent of the selection for port 2.

Baud Rate Switch

Three data signals are coded by the BAUD RATE SELECTION rotary switch on the two port mux box to select the baud rate shown in table II-6-1. The baud select receiver is selected by function decoder 1 and Memory Request and Read signal.

Port Options Switch

Three data signals are coded by the PORT OPTIONS rotary keylock on the two port mux box to select port options shown in table II-6-2. The port select receiver is selected by function decoder 1 and Memory Request and Read signal.

TABLE II-6-1. BAUD RATE SELECTION SWITCH

Baud Rate	Bit		
	2	1	0
110	1	1	1
300	1	1	0
600	1	0	1
1200	1	0	0
2400	0	1	1
4800	0	1	0
9600	0	0	1
19200	0	0	0

TABLE II-6-2. PORT OPTIONS SWITCH

Option	Bit		
	2	1	0
Disabled	1	1	1
MSG Only	1	1	0
DS Enabled	1	0	1
DS & Pwr Enabled	0	1	1
1 = TTL one (high)			
0 = TTL zero (low=ground)			

The signals from the switches are available to TPM hardware only. PP cannot read the signals.

Switch functions are described as follows:

Disabled	Port is disabled for all input and output operations. No data can be sent; all incoming signals are ignored.
MSG Only	Port is enabled to transmit for messages or diagnostics and other system originated output or input operations.
DS Enabled	Port is enabled to transmit for all functions including remote deadstart.
DS & PWR Enabled	Port is enabled to transmit for functions which are system originated, for remote power control and for remote deadstart.

RS-232-C SIGNALS

The following signals are available on each port (DST 2.1 and 2.2):

Transmitted Serial Data
Data Terminal Ready
Request To Send
Received Serial Data
Data Set Ready
Clear To Send
Carrier On
Ring Indicator

EXTERNAL DEVICE TO PP DATA FLOW

The following paragraphs describe the signal flow from the RS-232-C interface to channel 15. Figure II-6-2 shows the RS-232-C protocol.

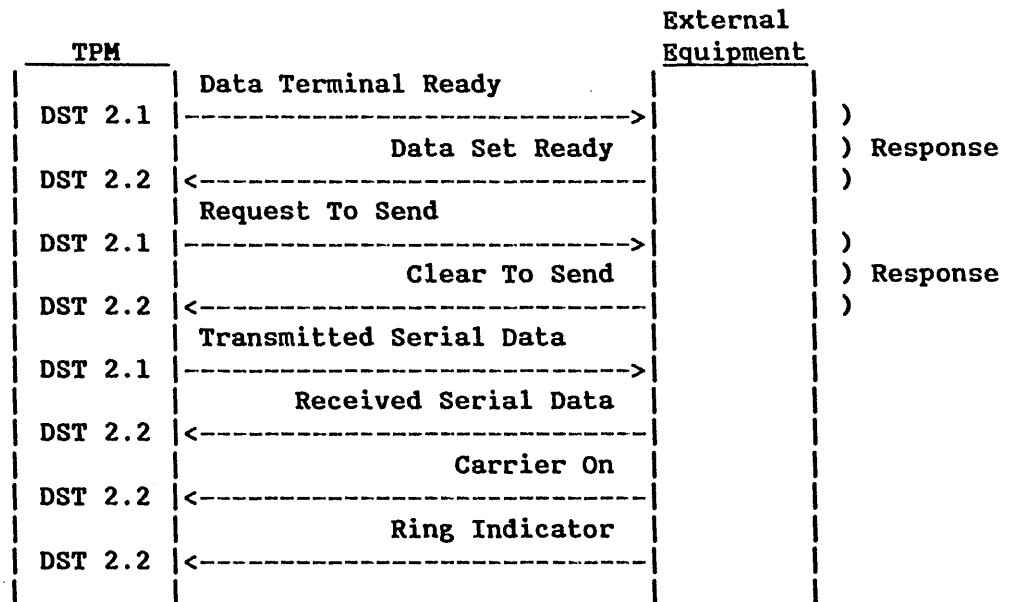


Figure II-6-2. RS-232-C Protocol

The voltage level converters (DST 2.2) change the RS-232-C interface signals into TTL levels for input into the UART (DST 2.1). The UART converts the serial data into 8-bit parallel data on the microprocessor bidirectional bus. The data feeds the PIO port 2 (DST 2.1) which feeds the TTL to ECL converter (DST 2.2). The ECL data is clocked into the channel 15 data register port 0 and 1 (DST 2.10) by the function decode circuit (DST 2.4).

The PIO port 2 data is strobed through the port 0 and 1 status registers (DST 2.2) to two of the inputs of the TPM. The other two inputs receive the status of the ports. The data is selected by the channel 15 function codes of 00XX read status or 01XX read data, and decoded in the channel 15 function decode circuit (DST 2.4).

The data is clocked into the channel 15 data register (DST 2.10). The register loads the data onto the channel 15 bus driver (DST 2.10) which drives the barrel 0, 1 channel bus.

The RS-232-C signal definitions are as follows:

Transmitted Serial Data	Serial data transmitted from TPM to external data terminal equipment.
Data Terminal Ready	Indicates a request by TPM to connect to external data terminal equipment.
Request To Send	Indicates a request by TPM for external data terminal equipment to prepare for data transmission from TPM.
Received Serial Data	Serial data transmitted from external data terminal equipment to TPM.
Data Set Ready	Indicates to TPM that external data terminal equipment is connected and ready for use. The signal is a response by external data terminal equipment to Data Terminal Ready signal from TPM.
Clear To Send (Data Carrier Detector)	Indicates to TPM that external data terminal equipment is ready to receive data. The signal is a response by external data terminal equipment to Request To Send signal from TPM.
Carrier On	Indicates to TPM that external data terminal equipment (modem) is receiving a suitable signal.
Ring Indicator	Indicates to TPM that external data terminal equipment (modem) is receiving a ringing signal on its communication channel; not disabled by off condition of Data Terminal Ready signal.

RS-366 INTERFACE

The following signals are available:

Call Request)
Digit Present) DST 2.3
Four Data Bits)
Power Indication)
Data Line Occupied)
Present Next Digit) DST 2.2
Abandon Call)
Call Origination Status)

Definitions of the foregoing signals are given in the following paragraphs:

Call Request

The TPM generates this signal to request the automatic calling equipment to originate a call. This signal is maintained in the on condition until the Call Origination Status signal is turned on.

Digit Present

The TPM generates this signal to indicate that the automatic calling equipment may read the data bits.

Four Data Bits

The TPM generates these four binary data signals as a code to the automatic calling equipment shown in table II-6-3.

TABLE II-6-3. DIGIT CODE OF RS-366 DATA BITS

DIGIT	DATA SIGNALS			
	NB8	NB4	NB2	NB1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
*	1	0	1	0
#	1	0	1	1
End of Number	1	1	0	0
* SEP	1	1	0	1
Unassigned	1	1	1	0
Unassigned	1	1	1	1
* SEP Separator Digit				

Power Indication

The automatic calling equipment generates this signal to indicate to the TPM that power is on in the automatic calling equipment.

Data Line Occupied

The automatic calling equipment generates this signal to indicate to the TPM that the communication channel is in use.

Present Next Digit

The automatic calling equipment generates this signal to indicate to the TPM that the automatic calling equipment is ready to accept the next digit.

Abandon Call

The automatic calling equipment generates this signal to indicate to the TPM that the connection to a remote data station is probably not successful and suggests to the TPM to abandon the call. This signal by itself does not abandon the call. The automatic calling equipment may take up to 60 seconds before issuing this signal.

Call Origination Status

The automatic calling equipment generates this signal to indicate to the TPM that the automatic calling equipment has completed its call functions and that control of the communication channel is transferred from the RS-366 Interface to the RS-232-C Interface. The modem carrier on the answering end triggers the call origination status in the automation calling equipment. The PP, however, should monitor this signal for at least 60 seconds.

PP TO EXTERNAL DEVICE DATA FLOW

The following paragraph describes the signal flow from the PP through channel 15 to the RS-232-C interface. A PP send 16 data bits onto channel 15. Only 8 bits of this data are loaded to the FIFO memory. Sixty-four PP words (16 bits) fill the FIFO memory. This data feeds the channel bus receiver (SCH 2.0). Data from either barrel 0 or 1 is selected by the channel data bus mux and fed to the channel data register. The data feeds the data select mux which selects the lower 8 bits (56 through 63) for output to the ECL to TTL converters (DST 2.3). The converters feed the 8-bit data to the FIFOs.

The format of the data from PP to TPM is as follows:

	56	57	58	59	60	61	62	63
WORD	FIRST DIGIT				SECOND DIGIT			
0	F	F	F	F	S	S	S	S
	THIRD DIGIT				FOURTH DIGIT			
1	T	T	T	T	f	f	f	f
	FIFTH DIGIT				SIXTH DIGIT			
2	fi	fi	fi	fi	S	S	S	S
	n-1 digit				n digit			
MAXIMUM 16	fi	fi	fi	fi	S	S	S	S

NOTES:

1. Last digit must be an EON (OC_H).
2. Maximum digits are 32 ($n=32$ maximum).
3. Any digit except the last can be a SEP (OD_H).
4. Digits must be valid (in range of 0-9 decimal).

Two separate FIFOs (DST 2.3) connect with each port of the TPM. The FIFOs are 8 bits wide and 64 words long. The serial type interfaces are extremely slow when compared with the PP speed. If there are no buffers, an output instruction from a PP causes the PP to hang for long periods while the slow RS-232-C Interface sends the data. The channel outputs data until the FIFO fills. When the FIFO is full, the hardware sends an Inactive back to the PP to inform the PP that no more data should be sent. The PP can carry on with other operations.

The condition of the FIFOs is monitored by the microprocessor. When the microprocessor reads the status of an FIFO and finds that the Output Ready signal is active, it sends a Function signal to PIO1 1 port A. PIO1 selects FIFO data from the TTL data mux and feeds the UART port 1 (DST 2.1).

The FIFOs are used solely for output operations from the PPs. The FIFO output is selected by the TTL data mux PIO 1 port A. This port outputs the data onto the microprocessor bidirectional data bus. The data is input to the UART (DST 2.1) which converts the 8-bit parallel data into serial data and feeds the RS-232-C Interface through the voltage level converters.

POWER CONTROL INTERFACE

The power control interface is a two-conductor BNC connector located on the cable connector panel. The BNC connector is connected to a mechanical relay (500 mA maximum contact current) on CK. The set power on relay contacts (DST 2.1) close when a power up command is given to TPM and open when a power down command is given to TPM.

CALENDAR CLOCK

The calendar clock (DST 2.0) operates on its own 32.768-kHz crystal. The accuracy of the clock depends on temperature and proper tuning of a trimer capacitor located on the CK pak at location IOU-20.

POWER CONTROL

The power control interface is a two-conductor BNC connector on the cable connector panel. The BNC connector is connected to a mechanical relay (500 ma maximum contact current) on CK. The relay contacts close when a power up command is given to the TPM and open when a power down command is given to the TPM. If the power control box is set up for remote power control, removing the CK logic board from the mainframe opens the circuit and the system powers down. To prevent the system from powering down, set the controls on the power control box to LOCAL before removing the CK logic board. The system also powers up or down without the power control box if the switch on the mainframe is in the REMOTE mode and the PORT OPTIONS switch is in the DS & PWR ENABLED position.

INTERNAL INTERFACE

The interface between channel 15 and the PPs is identical with all other PP to channel interfaces. All I/O instructions can be performed on channel 15. The internal channel data path is 16 data bits plus 1 parity bit. The PPM to TPM and TPM to PPM data flow is via barrel output mux (SCH 2.0) and channel 15 data input mux (SCH 2.0).

PP TO TPM FUNCTION CODES

The TPM uses the least significant 12 bits of data from channel 15 as the function code. A 12-bit function word from PP is translated to specify the operating condition of TPM as shown in table II-6-4. The TPM responds with an Inactive-In signal to any function code received from channel 15 as long as one of the two ports is selected. If the function code is not used, the TPM remains selected but is not set up for any operation.

TABLE II-6-4. PP TO TPM FUNCTION CODES

Code (octal)	Function
7XX0	Select port 0
7XX1	Select port 1
6XXX	Deselect TPM
04X0	Clear Data Terminal Ready signal
04X1	Set Data Terminal Ready signal
05X0	Clear Request To Send signal
05X1	Set Request To Send signal
07XX	Clear output and input buffers
1X02	Read Deadstart Port/Terminal Type
1X04	Read calendar clock
1X05	Write calendar clock
00XX	Read TPM status
01XX	Read port data
02XX	Write port data
03YY	Set port operation mode
1X06	Write auto-dial-out data
1X07	Read auto-dial-out status
1X10	Abandon call
NOTES	
X = Don't care	
In the least significant octal number of 7XX0, 7XX1, 04X0, 04X1, 05X0, and 05X1 function codes, only bit 63 is used; bits 62 and 61 are don't care.	
Y = 0,1	

The following describes the TPM function codes:

Select
(7XX0,7XX1)

Selects one of two ports; port number (0 or 1) is specified by the least significant bit (63) of the select function code. When a port is selected, all subsequent function codes and data are sent to that port only. If one port was previously selected and a select function code is issued to select the unselected port, the previously selected port is deselected and the previously unselected port is selected. The TPM always responds to the select function code with an Inactive signal to channel 15.

Deselect TPM (6XXX) Deselects any TPM ports selected from channel 15. After this function is issued, channel 15 may be used for PP to PP communication. Deselection of a port does not affect any operations between TPM and external equipment. For example, if an output operation (02XX function) is performed and the 64-character output buffer is full, TPM continues to empty the buffer even though the port is deselected.

Clear Data Terminal Ready (04X0) Clears Data Terminal Ready signal for the selected port. See RS-232-C Interface definition for description of this signal.

Set Data Terminal Ready (04X1) Sets Data Terminal Ready signal for the selected port. See RS-232-C Interface definition for description of this signal.

Clear Request To Send (05X0) Clears Request To Send signal for the selected port. See RS-232-C Interface definition for description of this signal.

Set Request To Send (05X1) Sets Request To Send signal for the selected port. See RS-232-C Interface definition for description of this signal.

Clear Output And Input Buffers (07XX) Clears output and input data buffers on the selected port.

Read Deadstart Port/Terminal Type (1X02) A PP uses this function to identify the port which initiated the last deadstart operation. Also identifies the class of terminal which is connected to that port. The function (1X02) is sent to the TPM by the PP and in response a word is returned for the PP to input. The format of the response word is:

56	57	58	59	60	61	62	63
T	T	T	T	P	P	P	P

Port Number	-	0	Channel 10
(bits 60-63)	1		Port 0
	2		Port 1
	3-F(16)		Not assigned
Terminal Type	-	0	CC545
(bits 56-59)	1		CDC752/CDC722 (Viking TTY)
	2		CDC721 (Viking X)
	3-F(16)		Not assigned

The TPM determines and stores the terminal type and port of the logged-in deadstart device. The actual implementation can be by automatic terminal detection or by directly inquiring the terminal operator.

Read Calendar Clock (1X04)	A TPM select function to port 0 or 1 is needed to read calendar clock. After channel 15 has sent this function and an Active-Out signal, TPM responds with eight full signals to channel 15, each accompanied by an 8-bit word of data in format shown in table II-6-5. After the eight words are sent, the TPM sends an Inactive. If this input sequence is terminated early with an Inactive signal from channel 15 (less than eight words read), effects of TPM on channel 15 are undefined. When bit 63 (LSB) is set, the wall clock time integrity is lost (that is, there has been a power failure and the clock data is no longer valid).
Write Calendar Clock (1X05)	A TPM select function to port 0 or 1 is required to write calendar clock. This function, followed by an Active-Out signal, tells TPM to treat channel 15 data-out as data to set the calendar time (see table II-6-6). All six words must be written each time the calendar clock is set. If this output sequence is terminated early with an Inactive-Out signal from channel 15 (less than six words written), the effects on TPM and calendar clock are undefined. The smallest time unit that can be set is the unit of minutes. Tenths of seconds, units of seconds and tens of seconds are set to zeros. Writing the wall clock automatically resets bit 63 of the status word to a zero value indicating wall clock data is now valid.
Write Port Data (02XX)	Requests a data output operation from TPM. Data output operation is initiated when channel 15 sends write port data function and an Active-Out signal. If output buffer of selected port is full, TPM responds with an Inactive signal to channel 15. If output buffer is not full, TPM sends an Empty signal to channel 15 for each Full signal received from channel 15 until output buffer becomes full. Full signal from channel 15 that fills the output buffer causes TPM to respond with an Inactive signal instead of an Empty signal. Each data word is eight bits (56 through 63).

The TPM sets Request To Send and Data Terminal Ready signals and begins transferring data from the output buffer to RS-232-C interface. Data transfer starts as soon as one character is in the output buffer and continues until the output buffer is empties, even though the port may be deselected. A clear output and input buffers function (07XX) terminates data transfer (see 07XX function). The Request To Send and Data Terminal Ready signals are cleared by TPM when the output buffer empties, unless they are previously set by a set data terminal ready function (04X1) and a set request to send function (05X1).

TABLE II-6-5. TPM READ CALENDER CLOCK

<u>Word</u>	<u>Status</u>				<u>Information</u>			
	56	57	58	59	60	61	62	63
0	0	0	0	0	0	0	0	S
	Tens Of Years				Units Of Years			
1	Y	Y	Y	Y	y	y	y	y
	Tens Of Months				Units Of Months			
2	0	0	0	M	m	m	m	m
	Tens Of Days				Units Of Days			
3	0	0	D	D	d	d	d	d
	Tens Of Hours				Units Of Hours			
4	0	0	H	H	h	h	h	h
	Tens Of Minutes				Units Of Minutes			
5	0	M	M	M	m	m	m	m
	Tens Of Seconds				Units Of Seconds			
6	0	S	S	S	s	s	s	s
	Reserved For Future Use							
7	0	0	0	0	0	0	0	0

TABLE II-6-6. WRITE CALENDER CLOCK

<u>Word</u>	<u>Status</u>				<u>Information</u>			
	56	57	58	59	60	61	62	63
	Tens Of Years				Units Of Years			
0	Y	Y	Y	Y	y	y	y	y
	Tens Of Months				Units Of Months			
1	0	0	0	M	m	m	m	m
	Tens Of Days				Units Of Days			
2	0	0	D	D	d	d	d	d
	Tens Of Hours				Units Of Hours			
3	0	0	H	H	h	h	h	h
	Tens Of Minutes				Units Of Minutes			
4	0	M	M	M	m	m	m	m
	Reserved For Future Use							
5	0	0	0	0	0	0	0	0

Read TPM
Status (00XX)

Requests TPM for an RS-232-C port status operation. After channel 15 sends this function and an Active-Out signal, TPM responds with one Full signal accompanied by a status word shown as follows:

<u>Bit</u>	<u>Description</u>
52-57	Not used (zero)
58	Call origination status (used on port 1 only for auto-dial-out)
59	Output buffer not full
60	Input ready
61	Carrier on
62	Data set ready
63	Ring indication

Output buffer not full indicates the 64-character output buffer for the selected port has less than 64 characters in it. FIFO is not full.

Input ready indicates the selected port has data ready to input to a PP.

For a description of the remaining status bits, see paragraph titled RS-232-C signals above.

Read Port
Data (01XX)

Requests TPM for a data input operation. After channel 15 sends this function and an Active-Out signal, TPM responds with only one Full signal accompanied by a data word shown as follows:

<u>Bit</u>	<u>Description</u>
52	Data set ready
53	Data set ready and carrier on
54	Data-in overrun
55	Data-in framing or parity error
56-63	Data-in bits

Data set ready (DSR) sets when the Data Set Ready signal from the terminal is in the ON condition.

Data set ready (DSR) and data carrier detector (DCD) set when both Data Set Ready and Carrier Detector signals from the terminal are in the ON condition.

Data-in overrun indicates the selected port has lost input data due to data coming in faster than the PP takes it. The data that accompanied this bit is the last data that TPM received and is written over the original data in the input buffer.

Data-in framing or parity error indicates selected port has detected a parity error or a framing error (invalid stop bits) on data received from an external device.

Data-in bits is data received on selected port when the input ready status bit (bit 60) is set. These bits are undefined when input ready is not set.

**Set TPM
Operation
Mode (03YY)**

Sets port operation mode register and specifies TPM operation mode according to bits 58 through 63 as follows:

<u>Bit</u>	<u>Description</u>
58	Enable loop back
59	Disable parity bit
60	Select number of stop bits
61, 62	Select number of bits/character
63	Select odd/even parity mode

Enables loop back, when set, enables a round trip data path from channel 15 to selected RS-232-C port (UART chip) and back to channel 15. When in loop back mode, UART chip does not transmit data externally.

Disable parity bit, when set, no parity bit is transmitted on the selected RS-232-C port. Parity checking on input data is disabled. The stop bit(s) immediately follow the last data bit.

Select number of stop bits selects number of stop bits. When clear one stop bit is used. When set, two stop bits are used. Select number of bits/character are shown as follows:

<u>Bit 61</u>	<u>Bit 62</u>	<u>Bit/Character</u>
0	0	5
0	1	6
1	0	7
1	1	8

Select odd/even parity mode selects type of parity to be transmitted and type of parity expected in input data. When set, even parity mode is selected. When clear, odd parity mode is selected.

Write Auto-
Dial-Out Data
(1X06)

This function is used to set up the TPM for an auto-dial-out operation. Port 1 must be selected before this function is issued. This function (1X06) followed by an Active-Out signal, sets up the TPM to treat channel 15 data-out as one digit to be passed on to the calling automatic equipment. See data format shown as follows:

<u>Bits</u>	<u>Description</u>
56-59	not used
60-63	number

The maximum number of digits that can be passed to the TPM is 32. An End of Number code must follow the last telephone number in the data. The End of Number code is in the four least significant bits of the last word and the four most significant bits are don't care. The End of Number code signals the TPM to initiate the auto-dial-out operation between the TPM and the automatic dialing equipment. The PP sets Data Terminal Ready on port 1 so that the automatic calling equipment can transfer control to the RS-232 Interface when the dialing is completed. While the TPM is dialing, the PP monitors the call by looking at the Abandon Call and Call Origination Status status bits.

If the PP determines that the call is to be abandoned, it must issue the function to reset Data Terminal Ready on port 1. The auto dialer must be equipped with the feature to terminate calls via Data Terminal Ready and not Clear Request.

This function should not be issued unless the auto-dial-out status bit Power Indication is a one and the auto-dial-out status bit Data Line Occupied is a zero.

Refer to data code table under the heading RS-366 Interface in this section.

Read Auto-
Dial-Out
Status (1X07)

This function requests the TPM for an auto-dial-out status operation. Port 1 must be selected before this function is issued. After channel 15 has sent this function (1X07) and an Active-Out signal, the TPM responds with only one Full signal accompanied by a status word shown as follows:

<u>Bits</u>	<u>Description</u>
52-59	Not used
60	Abandon call
61	Call origination status
62	Data line occupied
63	Power indication

For a description of these signals see the RS-366 Interface section.

Abandon Call (1X10)	This function tells the TPM to abandon the call being attempted. It is usually used when a PP has sensed the Abandon Call status bit during the dialing operation. The TPM clears the Call Request signal to the automatic calling equipment.
------------------------	---

PROGRAMMING CONSIDERATIONS

The TPM communicates, one at a time, with the two terminals connected to the external interface to establish communication between the PP and a terminal. Refer to tables II-6-7 and II-6-8 for output and input data procedures. A select function (7XXY) must be issued first and is formed from the least significant 12 bits of the channel 15 data word. The function specifies a select code to connect the TPM (7XXY) with the terminal the PP communicates with by setting or clearing bit 63 of the function code (7XXY).

When the connection is established, the TPM routes all data to the terminal designated by the select code. The TPM can accept a maximum data block length of 64 characters per terminal. During the block data transfer, the TPM terminates the output operation either when it receives an Inactive or when the output buffer is full. When the output buffer is full, the TPM sends an Inactive, instead of an Empty, to the channel in response to the last word output. This indicates the output buffer accepts the last output word but cannot accept more data from the PP.

Output to a full buffer is not allowed by the TPM. If an output is attempted (02XX function), the TPM causes channel 15 to appear and remain inactive after it receives an Activate from the PP.

Data is lost if the PP does not input the previous data before the new data arrives from the terminal. Input from an empty buffer is allowed by the TPM. However, the data received is undetermined.

Request To Send and Data Terminal Ready

The Request To Send (RTS) and Data Terminal Ready (DTR) signals are transmitted to the terminal automatically by the hardware under the following conditions:

- Data is in the UART output register.
- Data is in the FIFO output register.

The RTS and DTR signals can also be controlled by the 04XY and the 05XY function codes. The code sets these signals without regard to the UART or FIFO signals.

TABLE II-6-7. OUTPUT DATA PROCEDURE

Use the following procedure to output data to the terminal via the TPM:

1. Issue function 00XX to read summary status.
2. Activate channel 15.
3. Input one word.
4. Disconnect channel 15.
5. Check bit 59 of the status response to determine if FIFO is not full.
6. If FIFO is not full, issue function 02XX to output to FIFO.
7. Activate channel 15.
8. Output to FIFO until the word count equals zero (transfer complete) or until no more locations are available in FIFO.
9. If input ready bit is set, issue function 01XX to read terminal data.
10. Activate channel 15.
11. Input one word.

TABLE II-6-8. INPUT DATA PROCEDURE

Use the following procedure to input from the terminal via the TPM:

1. Issue function 00XX to read summary status.
2. Activate channel 15.
3. Input one word.
4. Disconnect channel 15.
5. Check bit 60 of status response to determine if terminal input ready bit is set.

EXTERNAL DEVICE TO TPM FUNCTIONS

The TPM monitors incoming signals and performs the functions described below. Two mechanisms alert TPM from an external device. The first mechanism is triggered by dialing the telephone number for the computer TPM which generates the Ring Indicator signal with auto-answer modem connected to the port.

The second mechanism is triggered by sending the ASCII code sequence for CNTL G (first alert signal - ASCII code=07h) and CNTL R (second alert signal - ASCII code=12h) to either port of TPM. The action taken by TPM depends on the position of the PORT OPTIONS switch on each port. The BAUD RATE SELECTION switches and the PORT OPTIONS switches are sampled by TPM when DEADSTART on the CC545 console is pressed or when a Ring Indication signal is received on either port 0 or port 1.

The PORT OPTIONS switch is also sampled when the code sequence for CNTL G is sensed on the input to the port. The PORT OPTIONS switch defines which external functions are enabled in TPM.

Any CRT terminals using the TPM for deadstart must be in page mode and have X-Y positioning enabled, even parity, 7 bits per character and 2 stop bits.

REMOTE DEADSTART

The remote deadstart feature can deadstart the PPs from a terminal on either port 0 or port 1 of the TPM. A deadstart password is required to use this feature on port 1. No security is provided for port 0. Refer to section II-2 for deadstart information.

AUTO ANSWER

- Ring Indicator signal is used for auto answer on both port 0 and port 1.
- Auto answer is supported if the 400-Hz power to the mainframe is on or off (60-Hz power must be on).
- Auto answer is disabled if the PORT OPTIONS switch is in the DISABLED position.
- Either the PPs or the TPM (microprocessor) may answer a call by setting DTR. If the PORT OPTIONS switch is in the MSG ONLY position, only the PPs answer. If the PORT OPTIONS switch is in the DS ENABLED or DS & PWR ENABLED positions, the PPs have first chance to answer. If the PPs do not answer within two rings, the TPM answers. The TPM then ignores any attempt by the PPs to communicate with the port which has the Ring Indicator signal until the TPM sends the OPERATOR ACCESS DENIED message.

REMOTE POWER CONTROL

Remote power control allows control of the main power for the system by commands from a terminal connected to port 0 or port 1 either directly or through a modem.

Security features which protect against invalid use of the remote power control feature are:

- The PORT OPTION switches on the mainframe must be set up. They are keylock switches where the key may be removed in any position. Only trained technical people have access to the switch panel located in the mainframe cabinet inside locked doors.
- On port 1 the user must enter a correct power control password (up to 15 characters long).
- Each time the Ring Indicator is sensed by the TPM on port 1, a new session is initiated and therefore password(s) must be entered by the user.
- If the password is not entered correctly in three attempts, the terminal is declared illegal and the connection to the terminal is dropped.
- On port 0 no password is required for remote power-on, but a password is required for remote power-off. Remote power-off always requires a password.
- The password may be entered or changed only from the CC545 console (if one is present) or the port 0 terminal. The password is entered with a command from the deadstart display.

PW PC XXX Set power control password where XXX is the password (1 to 15 characters long).

If the user is at a terminal using a modem to interface to port 1, remote power control is achieved by using the following command procedures. Note that in these sequences, OPR means Operator activities and TPM means Two Port Mux activities.

1. OPR Call the computer (Ring Indicator).
2. TPM Sense Ring Indicator signal. If power is on, wait approximately five seconds for the PPs to answer. See auto answer. If the PPs do not answer, set Data Terminal Ready and Request To Send and send message ENTER DEADSTART PASSWORD.
3. OPR Enter the deadstart password.
4. TPM Check for correct password. If it is correct, deadstart display is on.

If the power is off:

TPM Send message POWER IS OFF. DO YOU WANT POWER ON? (Y/N)

The TPM waits approximately 10 seconds for an operator response. If no response comes within the 10 seconds, the TPM clears the screen and terminates the session.

1. OPR Enter Y.

NOTE

If the operator enters anything other than Y, the TPM clears the screen and terminates the session.

2. TPM Send message ENTER POWER CONTROL PASSWORD.
3. OPR Enter the power control password.
4. TPM Send message POWER-UP SEQUENCE INITIATED. Deadstart display is on when power-up is complete.

NOTE

If the power-up is not completed within approximately 30 seconds, the TPM sends the message POWER IS STILL OFF, POWER UP SEQUENCE TERMINATED and terminates the session.

If a PP answers the call, the power is on. To turn the power off, a command from the deadstart display must be used as shown below.

1. OPR Press CNTRL and G code.
2. TPM Sense CNTRL and G code.

These codes are the first alert signals to the TPM. The TPM discontinues support of PP originated functions on both ports, sets the first alert flag and sends message OPERATOR ACCESS ENABLED. After the message has been sent the TPM waits approximately ten seconds for the second alert signal. If the second alert signal is not sensed in ten seconds, the TPM resets the first alert flag and continues support of PP originated functions to either port.

1. OPR Press CNTRL and R.
2. TPM Sense CNTRL and R code and take control of the port. PP originated operations on this port are discontinued. An ENTER DEADSTART PASSWORD message is sent to the terminal.
3. OPR Enter deadstart password.
4. TPM Check for correct password. If the correct password is entered, deadstart display is on.

NOTE

The TPM waits until after the last character of the message is sent before looking for the second alert signal. If any characters are received during the time interval when the message is sent, they are discarded.

1. OPR Enter OFF PWR.
2. TPM Send message ENTER POWER CONTROL PASSWORD.
3. OPR Enter the power control password.
4. TPM Check for correct password. If it is correct, open the power control relay and send message POWER-DOWN SEQUENCE INITIATED. When the power has dropped, send message POWER IS OFF.

The procedure for a terminal on port 0 is the same as for port 1 except no passwords are required for powering-up or getting the deadstart display. A password is still required for powering-down.

NOTE

If the user enters an invalid password, the TPM sends the message IMPROPER PASSWORD/TRY AGAIN. If the password is still wrong on the third try, the TPM sends the message OPERATOR ACCESS DENIED and clears Request To Send and Data Terminal Ready signals.

After the password is entered correctly, the TPM does not ask for a password until it senses the next Ring Indicator signal or the session is terminated with a QUIT command or a short or long deadstart.

If the operator tries to use a feature which is not enabled via the PORT OPTIONS switch, the TPM sends the message OPERATOR ACCESS DISABLED and continues support of PP originated functions to either port.

SECTION II-7

CC545 DISPLAY STATION CONTROLLER

=====

The display station controller (DSC) provides parallel communication capability between a peripheral processor (PP) and the CC545 Display Console. During deadstart the DSC provides communication capability between the microprocessor and the CC545 console. The DSC resides on CQ pak, which includes an internal input/output (I/O) channel. Connection to the CC545 console is via the channel 10* port.

The DSC generates displays in either dot mode for graphic displays, or character mode for alpha-numeric displays.

The DSC supports all CYBER 170 12-bit function codes (except channel pass-on and dual station features). It also supports CYBER 170 display code character set.

The DSC can accept 16-bit data (two ASCII characters) for input to the microprocessor. All unused characters are displayed as blanks. In keyboard input mode, the DSC translates the input character into ASCII code (if ASCII mode is selected).

INTERFACE DEFINITION

DSC TO PP

The display station controller (DSC) option is permanently wired to channel 10. If the option is not installed, a regular CYBER 170 channel can be substituted on channel 10. Inter-PP communication is available on channel 10.

Not only the signals between the DSC and the PP but also the protocol used are identical to the signals and protocol used on the other CYBER 170 channels in the IOU.

DSC TO CONSOLE

The connection between the DSC and the console is via two CYBER 170 I/O cables 19.8 meters (65 feet) long (standard and maximum), with 19 coaxial wires in each. All signals to and from the console are at TTL levels (0 volt for logic 0 and +3.0 volts for logic 1), with the exception of the X and Y analog signals used to generate characters. X analog and Y analog are differential signals with excursions from +0.2 volt to +2.0 volts. The signals are listed in table II-7-1.

* All channel numbers are in octal.

TABLE II-7-1. DSC-CONSOLE SIGNALS

Signal	Number of Bits	MLBD Ref
DSC to Console		
X coordinate (X0-X8)	9	DSC 2.2
Y coordinate (Y0-Y8)	9	DSC 2.2
Character size (small, medium)	2	DSC 2.1
X analog (+,-)	2	DSC 2.4
Y analog (+,-)	2	DSC 2.4
Unblank left screen	1	DSC 2.4
Unblank right screen	1	DSC 2.4
Left or right screen select	1	DSC 2.1
Astigmatism	1	DSC 2.3
Console to DSC		
Keyboard data bits (58-63)	6	DSC 2.0
Keyboard up	1	DSC 2.0
Keyboard down	1	DSC 2.0
Console deadstart	1	DSC 2.0

The software displays data for both left and right screens simultaneously. Unblank left screen and unblank right screen are both transmitted to the console where a switch determines which presentation is displayed.

DSC TO MICROPROCESSOR

The DSC communicates to the microprocessor only during deadstart. Pressing the DEADSTART pushbutton on the console resets the microprocessor. The microprocessor runs programs which initialize the two port multiplexer hardware, displays the initial deadstart display on the console, and stops the program waiting for keyboard input. The signals are listed in table II-7-2. Refer to section II-2 for further deadstart information.

OPERATING MODES

12-BIT MODE OPERATION

A 12-bit function word from a PP is translated in the function register to specify the operation of the DSC as shown in table II-7-3.

PP sends a function word (bits 52 through 63) via channel 10 to DSC, where bits 52 through 54 branch to the display code decoder for equipment select and

bits 55, 57 and 59 through 63 go through the data source select mux to the function register (DSC 2.0). If bits 52 through 54 are equal to 7 (connect DSC), an inactive (channel 10) response is generated to load the function register to specify the screen select (bits 55, 57), mode select (bits 59, 60), and character size select (bits 62, 63).

TABLE II-7-2. DSC-MICROPROCESSOR SIGNALS

Signal	Number of Bits	MLBD Ref
Microprocessor to Console		
Full pulse	1	DSC 2.1
Keyboard select	1	DSC 2.1
Medium character select	1	DSC 2.1
Select CC545	1	DSC 2.0
Master clear	1	DSC 2.0
Hold function on Channel 10	1	DSC 2.0
Z80 Data bits 56-63	8	DSC 2.0
Console to Microprocessor		
Keyboard data bits 56-63, Parity	8/1	DSC 2.0

TABLE II-7-3. 12-BIT FUNCTION CODES

Function	Code (52-63)	Bit(s) Set	Description
Equipment Select	7XXX	52-54	Connects DSC; without this, other functions do not receive an INACTIVE response from DSC
Screen Select	70XX	52-54	Display when left screen selected
	71XX	52-54, 57	Display when right screen selected
	74XX	52-54, 55	Display when either screen selected
Mode	7X0X	52-54	Character mode
	7X1X	52-54, 60	Dot mode
	7X2X	52-54, 59	Keyboard input request
Character Size	7XX0	52-54	Small (64 characters per line)
	7XX1	52-54, 63	Medium (32 characters per line)
	7XX2	52-54, 62	Large (16 characters per line)

Character Mode

Once the DSC is set in character mode, it responds to data received from the channel as follows:

- If the data code is a 6XXX, the rightmost 9 bits of the word specify the X coordinate of the CRT beam (000 through 777).

X coordinate is output from channel 10 data register (DSC 2.1), selected by data source select mux (DSC 2.0), and clocked into S register (DSC 2.2). The most significant three bits (55 to 57) feed the X counter (register) (DSC 2.2) and the next six bits (58 to 63) feed the X mux (DSC 2.2). The FULL (delay) signal and data code 6XXX (X coordinate signal) are ANDed together (DSC 2.1) producing the strobe X signal. The strobe X signal selects S register bits 58 through 63 from the X mux and clocks this data into the X register (DSC 2.0). Upon receiving the X register enable signal produced by strobe X ANDed with T10 clock, the combined nine bits feed the X transmitter which outputs to the display. The X coordinate is automatically incremented until the next 6XXX data code is received.

- If the data code is a 7XXX, the rightmost 9 bits of the word specify the Y coordinate of the CRT beam (000 through 777).

Y coordinate is output from channel 10 data register (DSC 2.1), selected by data source select mux (DSC 2.0), and clocked into S register (DSC 2.2). The least significant nine bits (55 through 63) are clocked into the Y register by strobe Y signal and feed the Y transmitter which outputs to the display. The Y coordinate in the Y register is retained until the next 7XXX data code is received.

- Any other data code is interpreted as two 6-bit character codes with the first character in the leftmost 6 bits and the second character in the rightmost 6 bits.

The character code is output from channel 10 data register (DSC 2.1), selected by data source select mux (DSC 2.0), and clocked into S register (DSC 2.2). The S register output feeds the character data mux (DSC 2.2). The first character (bits 52 through 57) is selected by select code 0, and the second character (bits 58 through 63) is selected by select code 1. These character bits are used to form part of the address and ROM enable signals to access the beam movement ROMs and unblank ROM (DSC 2.4).

The character generation logic uses the contents of these ROMs to produce the appropriate character generation codes for the desired character and to control the unblank signal following each character shown in table II-7-4.

TABLE II-7-4. CHARACTER GENERATION CODES

V1	V2	H1	H2	B	(Toggle beam unblank)	Function as Decoded by Movement Generators
0	0	0	0	1		Toggle blank/unblank state
1	0	0	0	0		Move beam vertically by 1 increment
0	1	0	0	0		Move beam vertically by 2 increments
1	1	0	0	0		Reverse vertical direction
0	0	1	0	0		Move beam horizontally by 1 increment
0	0	0	1	0		Move beam horizontally by 2 increments
0	0	1	1	0		Reverse horizontal direction

Character size determines the number of characters that can be displayed on the screen. Each screen is a 512 character by 512 character matrix.

Screen formats for the three character sizes are:

<u>Character Size</u>		<u>Characters/Line x Lines/Screen</u>
Small	8 x 8	64 x 64
Medium	16 x 16	32 x 32
Large	32 x 32	16 x 16

In character mode, the X coordinate is automatically incremented for every character in the increment character position adder and X counter (DSC 2.2) by 8 for small characters, 16 for medium, and 32 for large characters.

Function register bits 62 and 63 from DSC function select mux (DSC 2.1), are decoded into large, medium and small character size signals. These signals are transmitted to display console for display size control and are fed to the increment character position adder (DSC 2.2) for X coordinate incrementing.

The initial X coordinate is selected from the X mux (DSC 2.2) and clocked into the X register by X register enable signal. The X register feeds the X coordinate into the X transmitter (DSC 2.0) and the increment character position adder. The increment adder adds the X coordinate bits 58 through 60 to the character size bits and feeds the X mux with the incremented X coordinate. This incremented X coordinate is selected from the X mux when strobe X is inactive for any other data code but 6XXX. The timing chain decoder (DSC 2.3) and the associated circuitry generates an increment X signal at T=25 which is ANDed with T10 clock to produce an X register enable signal to clock the X mux data into the X register. The X coordinate is incremented every time for each character when inactive strobe X signal is received.

Dot Mode

Selection of dot mode causes DSC to respond to 6XXX (X position) and 7XXX (Y position) data codes by painting a dot on the CRT screen after the reception of each 7XXX code.

Dot mode signal from the DSC function mux is ANDed with Y coordinate (7XXX) and full delay signals in the display control translator to generate a display dot signal. This signal sets the display dot flip-flop (DSC 2.3) and is fed to the dot mode transmitter to output an astigmatism signal to the display console for dot mode control.

The display dot signal is also ANDed with a 400-nanosecond pulse generated from the timing network to gate the right/left unblank translator (DSC 2.4). The translator outputs right unblank and left unblank signals and transmits them to the display console for control.

Keyboard Input Mode

When in keyboard input mode, keydown signal from the console is fed to the keydown register and one shot circuit (DSC 2.0) to produce a load signal. This load signal enables keyboard data bits 58 through 63 from the console to be latched into the keyboard buffer register and causes keyboard full flip-flop to go full. The full signal and keyboard data bits form an address to output DSC data bits 48 through 63 from the display code to ASCII conversion ROM.

PP sends function bits 52 to 63 (7X2X) over channel 10, in which a keyboard input pulse (DSC 2.1) is generated when an inactivate signal from PP is received. This pulse sets the keyboard full register and is passed on to channel 10 control macrocell (refer to section IV for channel control macrocell CC 3.1 for description). The control signals output from macrocell enable the loading of DSC data bits 48 through 63 into the channel 10 data register and their routing to the PP.

A character is transmitted in the lower 6 bits of the channel word, with the upper bits set to zero. If the keyboard buffer register is not full, a 00 code is loaded.

16-BIT ASCII MODE OPERATION

Both PP and microprocessor communicate with DSC in 16-bit ASCII mode. However, function signals from microprocessor for display are decoded in deadstart pak (CK) before sent out to DSC for specific operation. Also, the microprocessor does not communicate with DSC in dot mode but only in character mode or keyboard input mode. Refer to table II-7-2 for these signals.

One 16-bit function word from the PP is translated into a function register to specify the operation of the DSC as shown in table II-7-5.

TABLE II-7-5. 16-BIT FUNCTION CODES

Function	Code (52-63)	Bit(s) Set	Description
Equipment Select	0EXX	52-54	Connects DSC. Without this, no other function receives an Inactive response from the DSC.
Screen Select	0EOX	52-54	Display when left screen selected
	0E4X	52-54, 57	Display when right screen selected
	0FOX	52-55	Display when either screen is selected
Mode	XEO4	52-54, 61	Character Mode
	0EX8	52-54, 60	Dot mode
	0E14	52-54,59,61	Keyboard input to 8 bit ASCII character.
Character size	0EO4	52-54, 61	Small (64 charcters per line)
	0EO5	52-54,61,63	Medium (32 characters per line)
	0EO6	52-54,61,62	Large (16 characters per line)

Character Mode

Once the DSC is set in ASCII character mode, it responds to data received from the channel or microprocessor as follows:

- If the data code is a CXXX or DXXX, the rightmost 9 bits of the word specify the X coordinate of the CRT beam (000 through 1FF).
- If the data code is a EXXX or FXXX, the rightmost 9 bits of the word specify the Y coordinate of the CRT beam (000 through 1FF).
- Any other data code is interpreted as two 8-bit ASCII character codes with the first character in the leftmost 8 bits (bits 48 to 55) and the second character in the rightmost 8 bits (bits 56 to 63). See table II-7-6 for the ASCII character and displayed character and their respective codes.

The control signal select CC545 (from microprocessor) determines whether channel 10 (PP) data or microprocessor data is selected by the data source select mux (DSC 2.0). Upon receiving a full signal, this selected data is clocked into the S register and feeds the character data mux (DSC 2.2). Select code 2 selects the first character (bits 48 through 55 from PP); select code 1 selects the second character (bits 56 through 63 from PP); select code 3 selects microprocessor data bits 56 through 63. These selected character data bits are translated to the 6-bit display code in the ASCII to display code conversion ROM before they go to the character generation logic to produce the appropriate alphanumeric character.

ASCII Character		ASCII Code (Hex)		Display Code (Octal)	Character Display In Console
A	a	41	61	01	A
B	b	42	62	02	B
C	c	43	63	03	C
D	d	44	64	04	D
E	e	45	65	05	E
F	f	46	66	06	F
G	g	47	67	07	G
H	h	48	68	10	H
I	i	49	69	11	I
J	j	4A	6A	12	J
K	k	4B	6B	13	K
L	l	4C	6C	14	L
M	m	4D	6D	15	M
N	n	4E	6E	16	N
O	o	4F	6F	17	O
P	p	50	70	20	P
Q	q	51	71	21	Q
R	r	52	72	22	R
S	s	53	73	23	S
T	t	54	74	24	T
U	u	55	75	25	U
V	v	56	76	26	V
W	w	57	77	27	W
X	x	58	78	30	X
Y	y	59	79	31	Y
Z	z	5A	7A	32	Z
0		30		33	0
1		31		34	1
2		32		35	2
3		33		36	3
4		34		37	4
5		35		40	5
6		36		41	6
7		37		42	7
8		38		43	8
9		39		44	9
+		2B		45	+
-		2D		46	-
*		2A		47	*
/		2F		50	/
(28		51	(
)		29		52)
=		3D		54	=
,		2C		56	,
.		2E		57	.

Note: Except for five keyboard input characters described in table II-7-7, all ASCII characters not listed are displayed as blanks.

Dot Mode

Selection of dot mode causes the DSC to respond to CXXX or DXXX (X position) and EXXX or FXXX (Y position) data codes by painting a dot on the CRT screen after the reception of each EXXX or FXXX code.

Keyboard Input Mode

For the function OE14, the 6-bit display code from console is translated into 8-bit ASCII in the display code to ASCII conversion ROM (DSC 2.0) before latching into either channel 10 data register to PP (DSC 2.1) or keyboard data register to microprocessor (DSC 2.0). If the data is transmitted to the PP, the resulting character is the rightmost 8 bits of the channel word.

Table II-7-7 indicates the ASCII codes generated for several special characters from the console.

TABLE II-7-7. KEYBOARD INPUT CHARACTERS AND ASCII CODE

<u>Keyboard Character in Console</u> (Octal)	<u>Corresponding ASCII Code</u> (Hex)
53 Left blank	19
55 Right blank	15 NAK
60 Carriage Return	0D Carriage Return
61 Backspace	08 Backspace
62 Space	20 Space

PERFORMANCE CHARACTERISTICS

In dot mode, the DSC waits 3.3 microseconds following a 7XXX (12-bit mode) and EXXX or FXXX (16-bit mode) data code (Y coordinate) to allow the beam to settle and then unblanks the beam for 400 nanosecond to generate a dot. The total time between issuing a Y coordinate at the PP and receiving an EMPTY response from the DSC is 5.8 microseconds.

In character mode, each character requires 2.4 microseconds to form on the screen. In addition, each character time is preceded by a 1.2 microsecond delay to allow for beam settling. Total time between a FULL output from PP and an EMPTY reply from DSC is 7.3 microseconds.

SECTION III-1

INTRODUCTION

=====

Maintenance Access Hardware (MAH) is organized to perform the following operations:

- Initialize registers, controls, and memories.
- Monitor and record error information.
- Reconfigure system.
- Verify error detection and correction hardware.

MAH, as shown in figure III-1-1, consists of three major units:

- Maintenance Control Unit (MCU) which is a functional name given to the Peripheral Processor (PP) programmed to control various MAH operations.
- Maintenance Channel (MCH) which consists of an interface to the MCU and four radial interfaces (RI 0-3) to Maintenance Access Control (MAC).
- Maintenance Access Control (MAC) which is connected directly to one of four radial interfaces RI for each system element. The base system MAC (connected to RI 0/1) is an intelligent multiway data mux which operates under its own nanocode control. It responds to function words sent by MCH.

MAC connects the internal elements of the CPU through the maintenance channel to the IOU. This is controlled by 36-bit nanocode stored in ROMs. It also provides various data paths and controls to read or write CPU-0, CPU-1, IOU, and CMC registers.

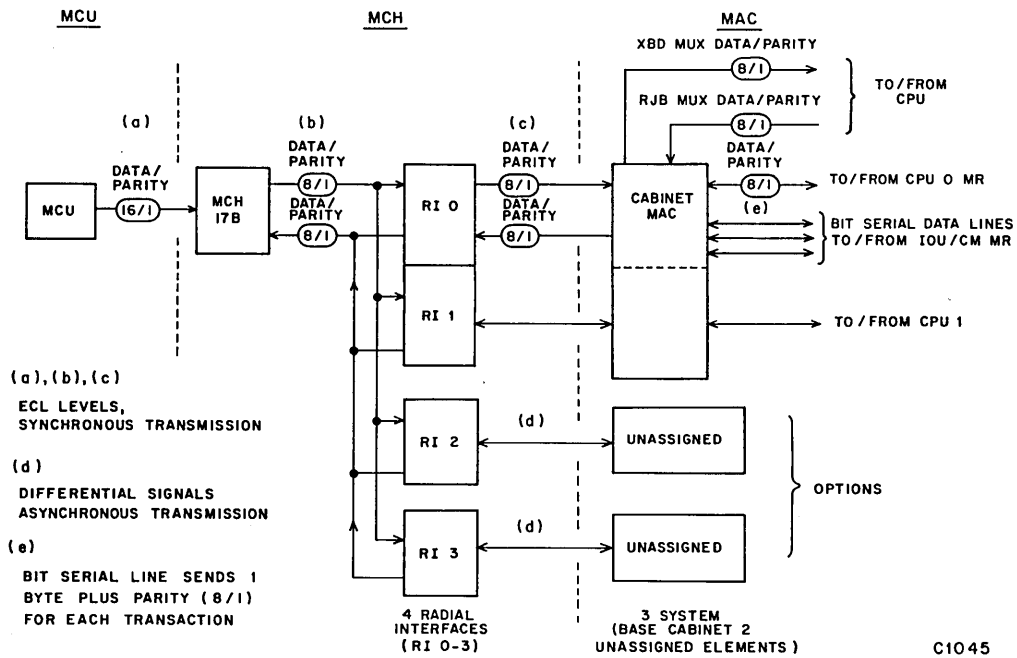


Figure III-1-1. Maintenance Access Hardware Organization

SECTION III-2

UNIT DESCRIPTION

=====

The three units MCU, MCH, and MAC are described here. Each system element has a MAC to perform MCH functions so only the base cabinet system MAC is described. Appendix D contains aids to assist your understanding of MAC/microcode operations.

MAINTENANCE CONTROL UNIT (MCU)

MCU is a functional name given to the PP programmed to control various MAH operations. Refer to the following for more details:

- PP Theory in part II in this manual.
- MCU Coding examples in section III-3.

MAINTENANCE CHANNEL (MCH)

All I/O instructions can be executed on MCH (formed on CJ pak). MCH interface to the PPs is the same as I/O Channels (on CH paks) with the addition of the time-out circuit (SCH 2.3). MCH, via radial interfaces 0 or 1 to MAC, is the internal interface between CPU 0, IOU, MEM, and CPU 1; whereas radial interfaces 2 and 3 are external interfaces connected to MACs of other mainframe elements. MCH selects one of four radial interfaces, controls the MAC dedicated to the selected radial interfaces, and buffers data communication between PPs and system elements.

The function word from MCU to MCH indicates the operation and the facilities involved. The function word uses the least significant 12 bits of MCH which are mapped as shown below.

48	51 52	55 56	59 60	63
Not Used	Connect Code	Op Code	Type Code	

The connect code, bits 52 through 55 (SCH 2.2), selects one of the radial interfaces and is not transmitted to MAC. MCH decodes the connect code as shown in table III-2-1. The system element remains connected to the interface until a function signal is received with a different connect code. The 8 least significant bits (56 through 63) are sent to radial interface with a function signal to establish the mode of operation in the selected device.

TABLE III-2-1. DECODING OF CONNECT CODE

Connect Code (Hex)	Radial Interface Selected	Assigned System Element
0	RI 0	Base System MAC
1	1	Base System MAC
2	2	Unassigned
3	3	Unassigned
4	N/U	
5	N/U	
6	N/U	
7	N/U	
8 through F	Inter-PP Communication	

TIME OUT

A time-out mechanism is provided in MCH (SCH 2.3). An active-out or a ready-out starts the time-out counter and a ready-in or an inactive-in resets it. If no response is received from the connected device in approximately 100 milliseconds interval, in the event of hardware malfunction in the MAC of the selected mainframe element, the time-out mechanism unhangs the PP. Connect codes

from 8 through F disable the time-out counter when the MCH is being used for inter-PP communication. The time-out counter is enabled on N/U (4 through 7) connect codes.

MCH INTERFACE TO PP

This interface (SCH 2.0) is identical with all other channel-to-PP interfaces. Data path is 16 bits plus parity. All I/O instructions can be performed using MCH. Error and channel flags (as on all other channels) are available on MCH. Timing and programming considerations are the same as for all other channels. Data parity is generated for the eight bits (56 through 63) that go to the radial interface.

MCH INTERFACE TO MAC (RADIAL INTERFACES)

This interface consists of four radial interfaces (SCH 2.2) numbered RI 0 through 3. All communication between MCH and MAC is on a ready-resume basis (handshake) through the unidirectional signals defined in table III-2-2.

The base system contains two internal radial interfaces: RI 0 is for IOU, CM and CPU 0; RI 1 is for CPU 1 of a dual CPU system that is enclosed in the same cabinet. Optionally, two external interfaces (RI 2 and 3) are available, which interface with 61-pin connectors. They are designed so that any element may be removed or shut down without affecting other elements.

TABLE III-2-2. SIGNALS BETWEEN MCH AND RI/MAC (Sheet 1 of 2)

Signal Name Number of Lines	Signal Names and Description
Data-in 8/1	Lines carry data from MAC to MCH, accompanied by a parity bit (SCH 2.2).
Data-out 8/1	Lines carry data or function codes to MAC, accompanied by a parity bit (SCH 2.2).
Ready-out 1	Signifies valid data on data-out lines in an output mode or an acknowledgement of a data word from MAC in an input mode (SCH 2.4).
Active-out 1	Goes to MAC when MCH is activated by PP. On output mode, it informs MAC that PP is about to begin data transfer. On in-put mode (after output of desired registered address), it tells MAC to send data (SCH 2.4).
Inactive-out 1	Goes to MAC when MCH is deactivated (by PP); it signifies end of data transfer. It is not transmitted when time-out counter deactivates MCH (SCH 2.4).
Function-out 1	Goes to MAC when PP executes a function instruction on MCH. It signals MAC that data-out lines are carrying a function code establishing operation mode in MAC (SCH 2.4).
Ready-in 1	Goes from MAC to MCH. It signals either data on data-lines in input mode or acknowledges data word from MCH in an out-put mode (SCH 2.3).
Inactive-in 1	Goes from MAC to MCH to signal receipt of a function (SCH 2.3).
Error-in 1	Accompanied by either an inactive-in or ready-in signals MCH when MAC detects an internal error condition during data transfer (SCH 2.3).

TABLE III-2-2. SIGNALS BETWEEN MCH AND RI/MAC (Sheet 2 of 2)

Signal Name Number of Lines	Signal Names and Description
Radial Interface Number 1	Indicates which connect code (depending on whether CPU 0 or 1 is selected) is being used (SCH 2.2).
Z80 Connected 1	Indicates whether the function is from IOU or Z80 microprocessor (SCH 2.2).
Summary Status-in 1	Signifies MAC has an error bit set in its summary status byte and is a static level. It goes from MAC to radial interface. Radial interface resynchronizes signal to IOU clock system and passes it to IOU maintenance register. Summary status-in signals from various mainframe elements are ORed together and set bit 59 in IOU summary status byte (MCH 2.1).
Exchange Accept-in 1	Goes from CYBER 170 mode processor to radial interface. Signal is sent to IOU directly by backpanel wire because CPU and IOU are in the same cabinet. Therefore, this signal is not used in the radial interface.

External interface uses dc differential signals. It receives signals from MAC of external mainframe element asynchronous to the IOU clock. All lines are unidirectional. Data and parity lines are at static levels, while control signals sent from the radial interface to mainframe element MAC are in 50-nanosecond pulses. Control signals from mainframe element MAC are 50 to 100 nanoseconds wide (measured at MAC). All signals from radial interface to mainframe element MAC are resynchronized by mainframe element to its own clock system.

Voltages on the differential line are 0 and -0.8 volts. Connection to IOU is made either internally (backpanel to backpanel) with Berg contacts and housings or externally through a single shielded cable (3000 type) with a 61-pin connector. Length of the connecting cables must not exceed 60.96 meters (200 feet).

MAINTENANCE ACCESS CONTROL (MAC)

The base system MAC (connected to RI 0 and 1) is an intelligent multiway data mux under nanocode control which is formed on DQ pak (MAC 1.0). It consists of a data selector and distributor, maintenance channel interface, CPU interfaces (one for each CPU), priority and access control, and maintenance register interfaces to IOU, Central Memory and CPU(s). The base system MAC is really two logical MACs--one accessed using a connect code of zero to IOU, CM and CPU 0, and the other accessed using a connect code of one to CPU 1.

The major functions of MAC are as follows:

- Perform tasks required by the function codes received from the IOU radial interface.
- Provide MAC trap addresses which are used as CP microcode entry points when MAC requires the assistance of CP microcode to complete a sequence of operations.
- Provide a data path and control to load or store the CP control store.
- Control the logical operating environment of the CP by means of EC register.
- Check and collect parity conditions within the CP through the PFS register.
- Capture corrected errors which may occur within the CP through the MCEL register.
- Enable diagnostic control logic within the CP through the PTM register.
- Control the optional Performance Monitor Facility (PMF).
- Provide a data path and control to read or write various CP internal registers through the CPU bus.
- Provide a data path and control to read or write various IOU and CM internal registers through the IOU/CM bus.

MAC responds to function words sent by MCH. The format of the function word from MCH is shown below.

56	59	60	63
Op Code		Type Code	

The decodings of the op code and assignments of the type code are described in table III-2-3.

TABLE III-2-3. OP CODE FUNCTIONS AND TYPE CODE ASSIGNMENTS

Op Code	Function
0	Stop CPU
1	Start CPU
4	Read
5	Write
6	Master Clear
7	Clear Errors
8	Echo
C	Read Summary Status Byte
F	Deadstart CPU
Type Code	Description
0	IOU Registers
1	CPU Register File
2	CPU MAP
3	CPU AD Register
4	Maintenance Scan
5	CPU Control Store
E	CM Registers
F	CPU Registers (except mentioned above)

CPU INTERFACE

Each of the two CPU interfaces consists of the following signals:

CPU to MAC

- (8/1) Data/Parity from RJB mux (MAC 2.0)
- (1) CPU Request MAC (MAC 2.7)
- (1) Accept (MAC 2.4)
- (1) Ready (MAC 2.5)
- (4) Select MAC code (MAC 2.3)

MAC to CPU

- (8/1) Data/Parity to XBD mux in BDP (MAC 2.0)
- (1) Request control store microcode (MAC 2.4)
- (1) Accept (MAC 2.7)
- (1) Ready (MAC 2.5)
- (3) Type code (MAC 2.2)
- (1) Write (MAC 2.5)

CPU 0 or CPU 1 request MAC is a 100-ns pulse sent by the requested CPU when it wants to read or write a register. This sets the requested CPU request flip-flop (MAC 2.7). All CPU requests require nanocode. If MAC is not busy, an accept is sent to the requested CPU to clear the CPU request flip-flop and set the MAC busy flip-flop (MAC 2.7). The CPU request determines the nanocode entry point. The 4-bit select code (bits 27 through 30) of RGC field of CPU micrand is used directly to branch to the proper routine. The other signals

(request microcode, accept from CPU, write, and type code bits) are used when MAC needs microcode assistance to access a register for an MCH request.

If CPU microcode assistance is needed, MAC accepts the MCH request, sets the request control store flip-flop, and goes idle. The request control store signal causes the microcode to take a microdetour at the end of the current instruction to a routine to service the MAC request. The address of the routine is determined by the type code bits and the write signal. The CPU microcode sends an accept (clear MAC request) to MAC to clear the request control store flip-flop, (MAC 2.7) then sends a request to read the control word register which contains the number (address) of the register required. This is handled like any other CPU request. The microcode then sends a second request to transfer the data between CPU and MCH. During the time MAC is idle between requests, it may accept and execute a request from the other CPU. In this case, the communication between CPUs and MCH may be delayed a bit but will not be affected.

MAINTENANCE REGISTER INTERFACES

The maintenance register interfaces consist of two CPU bidirectional buses, eight to ten (depending on optional channel increment) bidirectional bit-serial data lines, and various multiplexor and register selection lines.

Both CPU bidirectional buses having eight data and one odd parity bit are connected to CPU 0 and CPU 1 separately (MAC 2.0). Each bidirectional bit-serial data line, together with load and write selection signals, form a bit serial interface to pak which contains IOU or CMC maintenance register data (MAC 2.6).

Tables III-2-4, III-2-5, and III-2-6 show the IOU, CM, and CPU registers, respectively, which are accessed through MAC.

TABLE III-2-4. IOU REGISTER READ/WRITE VIA MAC

Type Code	Rgtr. No.	Name	Unit	First Byte	Direct MAC R/W
0	00	Status Summary	MAC	7	R
0	10	EID	MAC	4	R
0	12	Options Installed	IOU	6	R
0	30	EC	IOU	5	R W
0	18	Mask	IOU	0	R W
0	21	Bounds	IOU	0	R W
0	40	Status	IOU	4	R
0	80	Fault Status 1	IOU	0	R W
0	81	Fault Status 2	IOU	4	R W
0	A0	Test Modes	IOU	7	R W
0	D0	PP Status	IOU	0	R
0	D1	PP Memory	IOU	4	R W
0	D2	Channel Status	IOU	7	R

IOU Status Summary can be read via MCH by READ function (OP CODE = 4) Type No. = 0, Register No. = 00, or by READ SUMMARY STATUS BYTE (OP CODE = C) Type No. = C.
Registers D0-D2 are special registers which can only be accessed by the deadstart microprocessor; any attempt to read or write them by a PP results in a dummy operation (zero read/nothing written).

TABLE III-2-5. CM REGISTERS READ/WRITE VIA MAC

Type Code	Rgtr. No.	Name	Unit	First Byte	Direct MAC R/W
E	00	Status Summary	MAC	7	R
E	10	EID	MAC	4	R
E	12	Options Installed	CM	3	R
E	20	EC	CM	2	R W
E	21	Bounds	CM	0	R W
E	A0	CEL	CM	0	R W
E	A4	UEL1	CM	0	R W
E	A8	UEL2	CM	0	R W
E	B0	Free Running Counter (FRC)	CM	0	W
0	D3	CM Reconfiguration Switch	CM	0	R W
0	D4	Oscillator Select	CM	0	R W

Any attempt to write the FRC causes it to reset to all ones, regardless of data sent to MAC.
D3 and D4 are special registers which can be accessed by the deadstart microprocessor only; any attempt to read or write them by a PP results in a dummy operation (zero read/nothing written).

TABLE III-2-6. CPU REGISTERS READ/WRITE VIA MAC (Sheet 1 of 2)

Type Code	Rgtr. No.	Name	Unit	First Byte	Direct MAC R/W
F	00	Status Summary	MAC	7	R
F	10	EID	MAC	4	R
F	11	PID	MAC	7	R
F	12	Options Installed	MAC	2	R
F	13	VMCL	SOFT		*
F	30	EC	MAC	0	R W
F	31	CS Address	CS	6	R W
F	32	CS Breakpoint	CS	6	W
F	3F	RF Dump Address	SOFT		*
F	40	P-Register	EXEC		*
F	41	MPS Pointer	SOFT		*
F	42	MCR	EXEC		*
F	43	UCR	EXEC		*
F	44	UTP	SOFT		*
F	45	STL	SOFT		*
F	46	STA	SOFT		*
F	47	Base Const	SOFT		*
F	48	PTA	SOFT		*
F	49	PTL	SOFT		*
F	4A	PSM	SOFT		*
F	50	MDF	SOFT		*
F	51	MDW	SOFT		*
F	60	Monitor Mask	EXEC		*
F	61	JPS Pointer	SOFT		*
F	62	SIT	SOFT		*
F	80	PFS	MAC	4	R W
F	90	Petry CEL	MAC	4	R W
F	93	MAP CEL	MAC	7	R W
F	A0	PTM	MAC	7	R W
F	C0	Trap Enables	EXEC		*
F	C1	Trap Enables	EXEC		*
F	C2	Trap Enables	EXEC		*
F	C3	Trap Enables	EXEC		*
F	C4	Trap Pointer	SOFT		*
F	C5	Debug Pointer	SOFT		*
F	C6	Keypoint Mask	SOFT		*
F	C7	Keypoint Code	SOFT		*
F	C8	Keypoint Class	SOFT		*
F	C9	PIT	EXEC		*
F	E0	CFF	EXEC		*
F	E1	CFF	EXEC		*
F	E2	OCF	EXEC		*
F	E3	OCF	EXEC		*
F	E4	Debug Index	SOFT		*
F	E5	Debug Mask	SOFT		*
F	E6	User Mask	SOFT		*
1	00	RFA/RFB	SOFT		*
2	00	MAP	EXEC		*
3	00	AD	MAC		*
4	00	M-Scan	MAC	0	Read Scan Data
5	00	Control Store	CS	0	R W

TABLE III-2-6. CPU REGISTERS READ/WRITE VIA MAC (Sheet 2 of 2)

NOTES

Registers marked * under Direct MAC R/W indicate microcode is required in CPU control store to access them. When using type codes 3, 4, and 5, register number must be 00_{16} . For type code 1, register number denotes starting address in register file; number of 64-bit words transferred is determined by microcode routine. For product-set microcode, this number is 32_{10} . MCU software must agree with microcode. For type code 2, uses a microcode routine register number to determine which file to read. Product set microcode uses following convention:

<u>Register</u>	<u>File</u>
0-3	Assoc 0-3
4-7	Validity 0-3
8-B	Real 0-3

Full file (16 64-bit words) is transferred.

SECTION III-3

OPERATIONS

=====

MCU, MCH, and MAC operations are described in this section. In addition, MCU coding examples are provided. MAC circuits and nanocode organization are described under MAC operations.

MCU AND MCH OPERATION

MCH communicates in turn with all devices connected to the radial interface in the following order:

1. MCH issues a function to establish communication. The function word is the 12 least significant bits in the MCH and specifies the following:
 - A connect code to select inter-PP communication or a particular radial interface (and hence the device connected to that radial interface MAC). Refer to table III-2-1 (SCH 2.2).
 - An op code to establish the mode of operation. Refer to table III-2-3 (MAC 2.1).
 - A type code, if applicable. Refer to table III-2-3 (MAC 2.2).

The receiving device responds with an inactive signal to acknowledge the receipt of any function word. If no receiving device is connected to the interface selected by the connect code, or if the connect code was for inter-PP communication, no inactive response occurs. Thus, the channel remains active and full. If the function code has a parity error, the device responds with an inactive-in and an error-in signal.

2. The start processor, stop processor, master clear, and clear error functions require no further action. The read, write, request summary status byte, and echo functions require the PP to activate the channel.

To a request summary status byte function, MAC responds with a summary status byte and a ready-in signal (after receipt of the active signal). The PP inputs the word on MCH (MAC 2.4).

If the function was either read, write, or echo, MAC of the selected mainframe elements waits for two control words (CW; two bytes) to specify the starting address. The PP outputs two CWs to MCH. The first CW specifies the most significant eight bits of the starting address and the second CW specifies the least significant eight bits of the starting address when op code = F (Deadstart CPU). For read or write, the first CW is ignored and the second CW contains the address of the register required. For echo, the first CW is ignored and the second CW contains the data to be echoed or returned to the PP (MAC 2.2).

After the PP has output two words, it deactivates the channel.

NOTE

Before deactivating the channel, the PP ensures that MCH is empty; otherwise the last word may be lost.

3. The PP then activates MCH. If the functions are read or echo, MAC of the selected device sends the first control word to MCH. When the transfer is complete, the PP deactivates the channel. The channel may be full after the input transfer is complete and before the deactivate instruction if no word length is specified to MAC, so MCH responds to the last empty sent out.

On a write function, a PP outputs data to MCH and deactivates the channel when the transfer is complete. After each transfer, the PP checks the error flag on the channel to ensure that the transfer is error free (SCH 2.3).

4. If inter-PP communication uses MCH, the PP has to ensure that all the mainframe elements connected to MCH are deselected. This is done by issuing a function with the connect code of a nonexistent radial interface: 10_8 to 17_8 .

MCU CODING EXAMPLES

1. Immediate operations

These operations require only a single function code.

<u>Master clear, element 1</u>	<u>Description</u>
0020 LDC 0#160	Load function word into A register Connect code = 1 (second CPU) Op code = 6 Type code = not used
00760 FAN MCH	Function (A) on MCH

The operation is identical for start, stop, and clear error with the appropriate op code.

2. Check MAC interface

This operation turns around an output word to verify that MAC Interface is functioning correctly.

<u>Echo check, element 1</u>	<u>Description</u>
00770 FNC 180,MCH	Function word (180) to Maintenance Channel Connect code = 1 Op code = 8 Type code = not used

00740 ACN MCH	Activate MCH
0014 LDN 0	Load first test CW; output first CW (not returned)
0020 LDC 0#55	Load second test CW
00720 OAN MCH	Output second CW
0066 FJM *,MCH	Loop on channel full
00750 DCN MCH	Deactivate channel
00740 ACN MCH	Activate channel
0014 LDN 4	Load channel byte count into A
0071 IAM BUFF,MCH	Read four copies of second CW
00750 DCN MCH	Deactivate channel

3. Read per control word 1 and control word 2

This operation allows MCU to read a maintenance register from a system element, read a process state register, or control memory from a processor.

Read Processor Fault Status from Element 1

Description

00770 FNC 140,MCH	Function word (140) to Maintenance Channel Connect code = 1 Op code = 4 Type code = not used
00740 ACN MCH	Activate MCH
0014 LDN 0	Load first test CW
00720 OAN MCH	Output first CW
0020 LDC 0#80	Load second CW
00720 OAN MCH	Output second CW
0066 FJM *,MCH	Loop on channel full
00750 DCN MCH	Deactivate channel
00740 ACN MCH	Activate channel
0014 LDN 8	Load channel byte count into A
0071 IAM BUFF,MCH	Read processor fault status register
00750 DCN MCH	Deactivate channel

4. Write per control word 1 and control word 2

This operation allows MCU to write a maintenance register of an element, or write a process state register or control store of a processor.

Write Micrand into Soft Control Store

Description

00770 FNC 055,MCH	Function word (055) to Maintenance Channel Connect code = 0 Op code = 5 Type code = 5
00740 ACN MCH	Activate MCH
0014 LDN 0	Load control words with beginning address in control store
00720 OAN MCH	Output first CW

00720 OAN MCH	Output second CW
0066 FJM *,MCH	Loop on channel full
00750 DCN MCH	Deactivate channel
00740 ACN MCH	Activate channel
0014 LDN 0#3F	Load channel byte count into A
0073 OAM BUFF,MCH	Write soft control memory from BUFF
0066 FJM *,MCH	Loop on channel full
00750 DCN MCH	Deactivate channel

5. Write memory bounds register

This operation allows MCU to write eight bytes of data from a specific location into memory bounds register.

00770 FNC 05E,MCH	Function word (05E) to Maintenance Channel
	Connect code = 0
	Op code = 5
	Type code = E
0014 LDN 21	Load A with bound register number (21)
00740 ACN MCH	Activate MCH
00720 OAN MCH	Output first CW (ignored)
00720 OAN MCH	Output second CW (Bound register number)
00750 DCN MCH	Deactivate MCH
0014 LDN 8	Load A with word (byte) count
00740 ACN MCH	Activate MCH
00730 OAN *,MCH	Output eight bytes from location *
00750 DCN MCH	Deactivate MCH

MAC OPERATION

OVERALL OPERATION

1. MAC receives 4-bit function and 4-bit type code on the data lines with function signal. MAC returns an inactive signal to MCH in response to the function signal and enables function decoder. The decoded function is loaded into the function register and type code is loaded directly into the type code register. Also, the radial interface number is loaded into the select CPU flip-flop.
2. The next step of operation depends on the function being received. Function is decoded to determine whether or not it requires nanocode. Functions that require nanocode cause the MCH request flip-flop to set. If MAC is not busy processing a request from either CPU, the MCH request is accepted. This causes the MCH request flip-flop to clear, the MAC busy flip-flop to set, and nanocode execution to begin. If MAC is already busy, the channel request is held and the function has to wait until the MAC busy flip-flop clears before executing.
 - Functions 0, 1, 6, and 7 are immediate. They do not require nanocode and so can be executed immediately after the function is decoded.

- Functions other than those listed above require nanocode and MAC sequences. The function decode is used to determine the initial nanocode entry; type code and encoded control word are used for a branch address to the proper nanocrand routine.
- The functions to read or write CPU registers (marked with an asterisk in table III-2-6) require CPU microcode assistance; MAC sets the request control store flip-flop and goes idle.

MAC CIRCUITS

Parity and Error Reporting

Parity is provided for MAC nanocode and for data transferred through the channel interface or the internal interface. When a parity error is detected in either the MAC nanocode or in the operation (function) code, an error signal is transmitted with an inactive signal. MAC then completes the channel protocol and inhibits the operation. When a parity error is detected in the channel data during echo operation, an error signal is transmitted with a ready signal and MAC completes the operation. When a parity error is detected in the channel/bus during write/read operation, an error signal is transmitted with a ready signal and MAC completes the operation (MAC 2.8).

Access Control (MAC 2.2)

On MCH read/write functions, the control word/type code combination is checked to see if the request is valid. The control word is decoded by a ROM to give one of the following signals: hard register, valid CM, valid IOU, and microprocessor write (access bits 0 through 3). The hard register bit is used with a type code of F and control store run to determine if microcode assistance is needed and available. All register numbers are assumed to be valid for type code F (CPU) and are referred to the microcode if not hard. The microcode determines which are valid.

The valid CM or IOU bits (access bits 1 and 2) are used with type codes 0 and E. The microprocessor write bit (access bit 3) is used with type code 0 and the Z80 connected signal for registers which only the microprocessor and not the PPs may write.

If the request is for an invalid IOU or CM register, or for a CPU hard register in which control store is not running, MAC responds with dummy operations: for a read, zeros are returned for each byte; for a write, data is accepted but not written anywhere. No error indication is given.

Priority (MAC 2.7)

Requests arriving at the same time (that is, when request flip-flops set at the same time) are handled with the following priority: CPU 0 first, CPU 1 second and MCH third. If MAC is busy on a CPU 0 request, and a MCH request comes along, the MCH request will be serviced next unless a CPU 1 request comes along before MAC goes busy on the MCH request.

Bit Serial Control (MAC 2.6)

MAC uses bit-serial interfaces to access registers in either IOU or CM as shown in figure III-3-1. Bit-serial data transfers between paks contain either IOU or CM maintenance registers. MAC is under control of nanocode, MAC timing chain (MAC 2.5), and bit-serial counter as shown in figure III-3-2.

Normally, hardware is forced to the nanocode routine for function of read or write maintenance registers. The start of the nanocode sequence also starts the MAC timing chain. A signal of start bit-serial transfer is generated at MAC timing chain N3 and nanocode bit ten which are both set (bit-serial operation). This signal enables the bit-serial counter and loads the MAC bit-serial shifter with parallel entry of byte code and write data from mux 2 (MAC 2.0).

At N3 time also, byte counter is updated and compared to the required byte transfer (MAC 2-3). If the two numbers are not equal, next nanocode address is fetched upon receiving the next start sequence signal for next byte transfer. If the two numbers are equal, MAC goes into idle mode after the byte transfer.

Usually, all paks contain CM/IOU maintenance registers which shift in ones in their respective bit-serial shifter. MAC decodes the output of nanocode bits 11-14 (pak select code) according to table III-3-1 or table III-3-2 to access the required pak. Then bit-serial data is sent out every bit-serial timer clock interval (50 ns) through the bidirectional bit-serial bus to the selected pak shifter.

At time 5, a load signal sent by MAC causes the bottom five bits of the input bit-serial data (byte select code) in the selected pak shifter to transfer to the byte code register. These bits are used to select a particular byte at this time, which is loaded into the pak shifter.

For a write operation, MAC continues to send bit-serial data between time 6 to time 14. At time 15(F) where transfer of one byte and parity is finished, a second load signal sent by MAC causes the parallel output data in the pak shifter to be written to the selected maintenance register byte, providing there is no parity error. MAC looks at the selected data line at time 16 to see if there is an error (the line should be zero for no error).

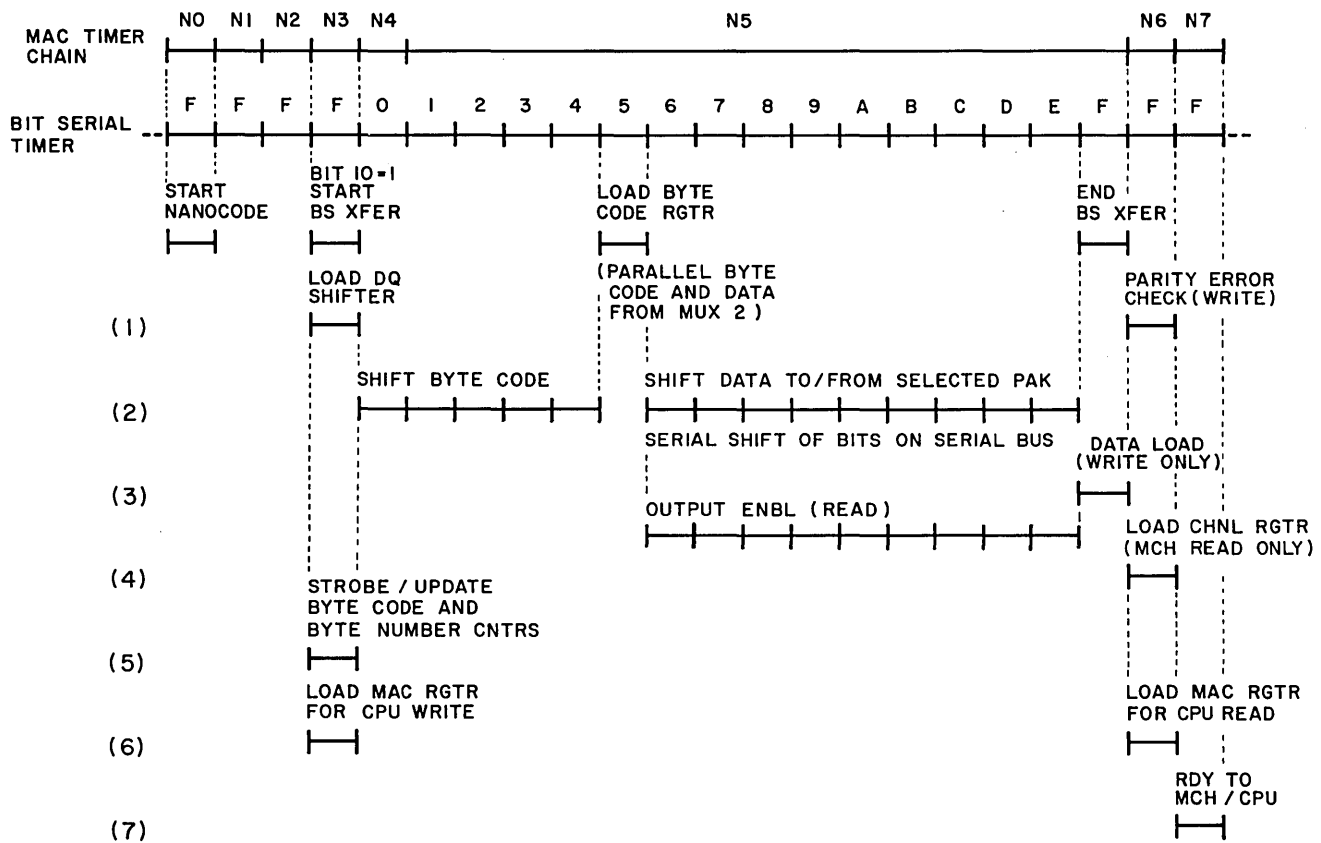
For a read operation, MAC sends a read signal to enable the output of the selected pak shifter to transfer bit-serial data into MAC bit-serial shifter. At time N6, received read data can transmit either to MCH or to CPU.

A ready signal to MCH or CPU at time N7 completes the byte transfer (read or write) and continues start sequence upon receiving next MCH fake ready or signal (MAC 2.4, 2.5).

Read or write CPU registers does not use bit-serial interface, except for the CPU environment control register and option installed register in DP pak.



Figure III-3-1. Bit-Serial Interface



C1044

Figure III-3-2. MAC Serial Timing

Table III-3-1. IOU/CM REGISTER BYTE FOR PAK SELECT 0 through 4

PAK SEL (nanocode bits SEL 11-14)	0	1	2	3	4
CODE (nanocode bits 26-30)	CJ	CH1	CH2	CH3	CH4
	RGTR CHAN BYTE	CHAN #	CHAN #	CHAN #	CHAN #
0	CHAN STATUS 10	CHAN STATUS 0	CHAN STATUS 6	CHAN STATUS 20	CHAN STATUS 26
1	STATUS 14	STATUS 1	7	21	27
2	STATUS 15	STATUS 2	—	22	30
3	STATUS 16	STATUS 3	11	23	31
4	STATUS 17	STATUS 4	12	24	32
5		STATUS 5	13	25	33
F	PP STATUS 7				

NANOCODE ORGANIZATION

Nanocode provides a sequence of control signals to direct the transfer of data between maintenance registers and the maintenance channel or CPU. For sequencing, the nanocode can branch to any address in the nanocode ROMs (MAC 2.3) and to a location determined by outside factors such as function, type code, or control word contents.

Nanocode Format

A nanocrand is a 36-bit word of nanocode. It is divided into fields which control various parts of MAC as follows (MAC 2.3):

<u>Bit Positions</u>	<u>Function</u>
0-7	Branch address (BRA)
8,9	Branch address select
10	Bit serial
11-14	Operation code or IOU/CM pak sel (MAC 2.6)
15	DS select
16-18	Byte Counter number (END)
19	Load byte counter
20-22	Data Mux 1 select (MAC 2.0)
23-24	Data Mux 2 select (MAC 2.0)
25	Transmit to MCH
26-30	IOU/CM byte code (MAC sel code)
31	Short sequence
32	Ready to MCH
33	Ready to CPU
34	Write
35	Parity (even)

Table III-3-2. REGISTER BYTE FOR PAK SEL 5 through 7, E AND F

MAC SEL CODE	PAK SEL	5		6		7		E		F	
		DP		CN		CL		DA		DC	
		RGTR	BYTE	RGTR	BYTE	RGTR	BYTE	RGTR	BYTE	RGTR	BYTE
0				FSI	0	PP STATUS	4	OI	0	EC	0
1				FSI	1	STATUS	5	OI	1	EC	1
2				FSI	2	STATUS	6	OI	2	EC	4
3				FSI	3			WR RECONF SW		BOUNDS	0
4				FSI	4			BOUNDS	4	CEL	0
5		CPU0	OI 3	FSI	5			BOUNDS	5	CEL	4
6				FSI	6	PP MEM DATA	3	BOUNDS	6	UEL1	0
7				FSI	7	PP MEM DATA	4	BOUNDS	7	UEL1	4
8		CPU0	EC 0	OI	2			-		UEL1	5
9				OI	3			CEL	1	UEL1	6
A		CPU0	EC 2	OI	4			CEL	2	UEL1	7
B		CPU0	EC 3	OI	5			CEL	3	UEL2	0
C		CPU0	EC 4	OI	6	EC	4	-		UEL2	4
D				OI	7	EC	5	UEL1	1		
E		M1EC	2	TM	7	EC	6	UEL1	2		
F		WR OSC SEL		TM	6	EC	7	UEL1	3		
10		CPU1	EC 0	PP MEM ADRS	1	OSB	0	-			
11				PP MEM ADRS	2	OSB	1	UEL2	1		
12		CPU1	EC 2			OSB	2	UEL2	2		
13		CPU1	EC 3			OSB	3	UEL2	3		
14		CPU1	EC 4	FS2	4	OSB	4				
15		CPU1	OI 3	FS2	5	OSB	5				
16		M1EC	6	FS2	6	OSB	6				
17				FS2	7	OSB	7				
18				MASK	0						
19				MASK	1						
1A				MASK	2						
1B				MASK	3						
1C				MASK	4						
1D				MASK	5						
1E				MASK	6						
1F				MASK	7						

Nanocode Map

The nanostore is organized as follows:

<u>Address</u>	<u>Nanocrand Function</u>
01-0A	Starting nanocrands of IOU register read sequences, with 01-0A being the entry points for individual sequences
1F	Request microcode assistance (for RF read)
20-27	Initial entry points forced by the hardware for a particular sequence
2F	Request microcode assistance (for MAP read)
3F	Read CPU AD register
4F	Start reading maintenance scan data
5F	Start reading CS
61-69	Starting nanocrands of CM register read sequences, with 61-69 being the entry points for individual sequences
71-7F	Starting nanocrands of CPU register read sequences, with 71-7F being the entry points for individual sequences
81-8A	Starting nanocrands of IOU register write sequences, with 81-83 being dummy write and 84-8A being the entry points for the individual sequences
9F,AF,BF	Dummy write for writing RF, MAP and CPU AD register respectively
CO-CF	Starting nanocrands for CPU requests
DF	Start for writing CS
E1-E9	Starting nanocrands for CM register write sequences, with E1-E3 being dummy write and E4-E9 being the entry points for individual sequences
F4-FF	Starting nanocrands for CPU register write sequences

NANOCODE SEQUENCES

Entry Points

The function from the MCH is decoded in MAC function decoder (MAC 2.1) to determine whether it needs nanocode. For any function (legal or illegal) that requires nanocode, the hardware is forced to one of eight entry addresses on the nanocode ROM address lines according to its function as follow:

<u>Function</u>	<u>Entry Address</u>
Read	20
Write	21
Dummy (invalid) read	22
Dummy (invalid) write	23
Deadstart	24
Request Status Summary	25
Echo	26
AD (CPU request)	27

Addressing the Nanocrands (MAC 2.3)

Current nanocode bits 8 and 9 determine next nanocode address as shown:

<u>Bits</u>		<u>Next Nanocode Address</u>	<u>Description</u>
8	9		
0	0	<div style="display: flex; justify-content: space-between; width: 100%;"> 0 4 5 7 </div> <div style="border: 1px solid black; padding: 2px; display: flex; justify-content: space-between;"> 00100 Start Address (5-7) </div>	Entry address for function requires nanocode
0	1	<div style="display: flex; justify-content: space-between; width: 100%;"> 0 1 3 4 7 </div> <div style="border: 1px solid black; padding: 2px; display: flex; justify-content: space-between;"> BRA 0 TCD (1-3) CW ROM (4-7) </div> <div style="text-align: center; margin-top: 5px;"> \ / </div> <div style="text-align: center; margin-top: 5px;">DCT</div>	Read/write request from MCH
1	0	<div style="display: flex; justify-content: space-between; width: 100%;"> 0 3 4 7 </div> <div style="border: 1px solid black; padding: 2px; display: flex; justify-content: space-between;"> BRA (0-3) CPU Function(0-3) </div> <div style="text-align: center; margin-top: 5px;"> \ / </div> <div style="text-align: center; margin-top: 5px;">MAC Sel Code</div>	Any CPU request (refer to MAC sequences-- microcode required).
1	1	<div style="display: flex; justify-content: space-between; width: 100%;"> 0 7 </div> <div style="border: 1px solid black; padding: 2px; display: flex; justify-content: space-between;"> BRA 0 - 7 </div>	

NOTE

Start address (5-7)	Bits 5 through 7 are the encoded bits for the eight functions requiring nanocode.
BRA (0-7)	Bits 0 through 7 of current nanocode.
TCD (1-3)	Bits 1 through 3 of type code from MCH.
CPU Function (0-3)	Bits 0 through 3 of MAC select code from either CPU 0 or CPU 1.
CW ROM (4-7)	Bits 4 through 7 from control word ROM (see table III-3-3 for contents of ROM).

In the case of a read or write function, a combination of type code bits 1 through 3 and control word ROM bits 4 through 7 which is called the device control table (DCT) is used to branch from the entry nanocrand to the starting

nanocrand of the appropriate sequence. The DCT is acted as a seven bits offset from the beginning of the read/write block.

For dummy (invalid) read/write, deadstart, request status summary, and echo functions, their corresponding entry nanocrand branches to their specific routine. The entry nanocrand itself supplies the full branch address (BRA 0 through 7).

For AD register (CPU request) function, CPU function bits 0 through 3 (MAC SEL CODE) are appended to the higher order half-byte of branch address (BRA 0 through 3) of the beginning of the AD block to branch to the correct location.

TABLE III-3-3. CONTROL WORD MAPPING ROM

Address	Contents	Address	Contents
00	1	81	A
10	2	90	6
11	D	93	8
12	3	A0	4
18	8	A4	8
20	5	A8	9
21	7	B0	6
22	A	D0	10
30	5	D1	11
31	B	D2	12
32	C	D3	13
40	9	D4	14
80	6		

To avoid conflicts, the same number in ROM can be assigned to two or more registers only if they are being used with different type codes.

Addresses D0 D4 are for microprocessor read/write operation.

All unlisted locations contain zeros. The ROM is disabled for operations which will require microcode assistance giving an effective DCT OFF.

Table III-3-4 shows nonrelocatable starting addresses and the DCT for the registers to which MAC has direct access. DCTs are formed by taking the low order three bits of the type code (two bits for registers D0 through D4) as the upper hexadecimal digit (MSB = 0 or 1 for read or write), and a value in

CW ROM (bits 4 through 7). Starting addresses in parenthesis are reserved to provide a no-op in case someone tries to read/write a write only/read only register.

Table III-3-4. NONRELOCATABLE STARTING ADDRESSES

Register	Type Code	CW ROM ADDRESS	DCT	Starting Addresses (Nonrelocatable)
SS	0,E,F	00	01,61,71	01,61,71,(81,E1,F1)
EID	0,E,F	10	02,62,72	02,62,72,(82,E2,F2)
Op. Inst.	0,E,F	12	03,63,73	03,63,73,(83,E3,F3)
EC(IOU,CPU)	0,F	30	05,75	05,75,85,F5
EC (CM)	E	20	65	65,E5
IOU Status	0	40	09	09,(89)
FS1	0	80	06	06,86
FS2	0	81	0A	0A,8A
TM	0	A0	04	04,84
BR	0	20	07	07,87
MR	0	18	08	08,88
PP STATUS	0	D0	10	10,(90)
PP MEMORY	0	D1	11	11,91
CH STATUS	0	D2	12	12,92
BR	E	21	67	67,E7
CEL	E	A0	64	64,E4
UEL1	E	A4	68	64,E8
UEL2	E	A8	69	69,E9
FRC	E	B0	66	(66),E6
OscSel	0	D3	13	13,93
Config. SW	0	D4	14	14,94
P1 S	F	31	7B	7B,FB
BKPT	F	32	7C	7C,FC
PFS	F	80	76*	76,F6
Retry CEL	F	90	76*	76,F6
MAP CEL	F	93	78	78,F8
PTM	F	A0	74	74,F4
PID	F	11	7D	7D,(FD)
M-Scan	4	00	4F	4F,(CF)
CS	5	00	5F	5F,DF

* Registers accessed by the same nanocrands.

Microcode requests (starting addresses for microcode requesting functions) must be at locations 1F, 2F and 3F.

MAC OPERATION WITHOUT NANOCODE

Operations without nanocode assistance do not cause MAC to go busy. Also, they do not interfere with a CPU function being processed by MAC unless master clear function is issued to the CPU that is talking to MAC. The following MAC operations require no nanocode:

Stop CPU	Halts CPU 0 or CPU 1 after current instruction. Op code of next instruction is in its respective CS address register. If CPU is already in halt status, function has no effect. If CPU is running but not executing instructions (that is, not taking op codes from RNI), function has no effect and a master clear halts CPU (MAC 2.7, CS 2.0).
Start CPU	Starts CPU 0 or CPU 1 executing micrands at address in its respective control store S register (MAC 2.7, CS 2.0).
Master Clear	Sends a 100-ns pulse to selected element, per type code, to clear various control flip-flops (MAC 2.1). Master clear will not clear error logs or fault status registers (FSR).
Clear Errors	Sends a 100-ns pulse to selected element, per type code, to clear error logs (MAC 2.1).

MAC OPERATION WITH NANOCODE

Echo Tests MCH to MAC interface by echoing back whatever data is in second CW. Type code is not used. Sequence is as follows:

1. MCH: Function
MAC: Inactive
2. MCH: Activate
3. MCH: Ready (first CW)
MAC: Ready
4. MCH: Ready (second CW)
MAC: Ready (second CW now in Register)
5. MCH: Inactive
6. MCH: Activate (channel request flip-flop sets; MAC goes busy)
7. MAC: Ready (contents of CW on data lines)
MCH: Ready (received)
MAC: Ready
8. MCH: Inactive (MAC busy clears; MAC goes idle)

Nanocode is used for data transfer (step 7 is repeated for each byte). MAC busy flip-flop sets when nanocode is being used; any other function which requires nanocode must wait until MAC is not busy.

Read Reads register specified by type code and second control word. Sequence is same as for echo except at step 7, where content is one byte of register on data lines.

Write Writes register specified by type code and second CW.
Sequence is same as for echo, steps 1 through 6. The rest are:

7. MCH: Ready (data to be written is on data lines)
MAC: Ready (data has been written)
MCH: Ready (last byte)
MAC: Ready (last byte)
8. MCH: Inactive

Nanocode is used for data transfer (step 7; repeated for each byte being written to the specified register).

Read Status Summary Byte Reads one byte IOU SS Register. Type code and control words are not used. Sequence is:

1. MCH: Function
MAC: Inactive
2. MCH: Activate (channel request flip-flop sets causing MAC busy to set)
3. MAC: Ready (summary status byte on data lines)
MCH: Ready (received)
4. MCH: Inactive (MAC goes idle)

Deadstart CPU Deadstarts CPU 0 or 1 at its respective 13-bit CS address contained in the two control words. Type code must be F, otherwise results are undefined. Sequence is:

1. MCH: Function
MAC: Inactive
2. MCH: Activate (Channel request flip-flop sets causing MAC busy to set)
3. MCH: Ready (first CW on data lines)
MAC: Ready (first CW loaded into S register bits 0-4)
MCH: Ready (Second CW on data lines)
MAC: Ready (Second CW loaded into S register)
4. MCH: Inactive (MAC goes idle)

At step 1, a 100-ns master pulse is sent to the selected CPU. At step 4 a 50-ns start (selected) CPU pulse is sent. Therefore, this function is essentially a combination of master clear, write (S register) and start CPU functions.

MAC REQUESTS

MAC can request data transfer originating in one of the following:

MCU

MCU can make two types of MAC requests:

- Those which can be done entirely by MAC

- Those which require microcode (that is, require CPU processor intervention)

CPU

Requests to the CPU are for those operations which specify a source and/or destination register inaccessible to the MAC. In these cases, control is passed to the CPU which may then request the MAC to perform certain data transfers to and from the shared AD register in the process of carrying out the microcode requesting operation. Refer to figure III-3-3 for flow diagram format.

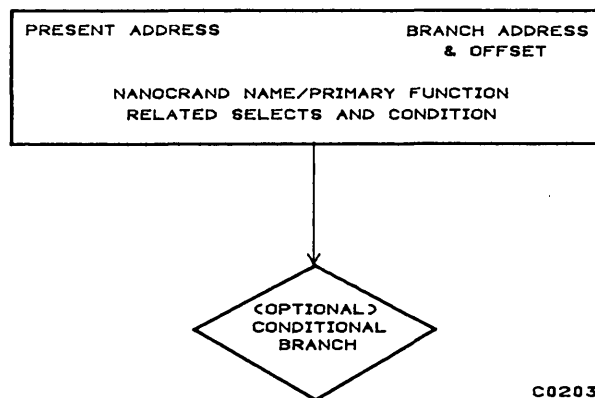


Figure III-3-3. Flow Diagram Format

Note that a branch address of next designates the next nanocrand in the sequence, although this may not be the next ROM location. Sequences do not necessarily appear in contiguous memory locations.

MAC SEQUENCES--NO MICROCODE REQUIRED

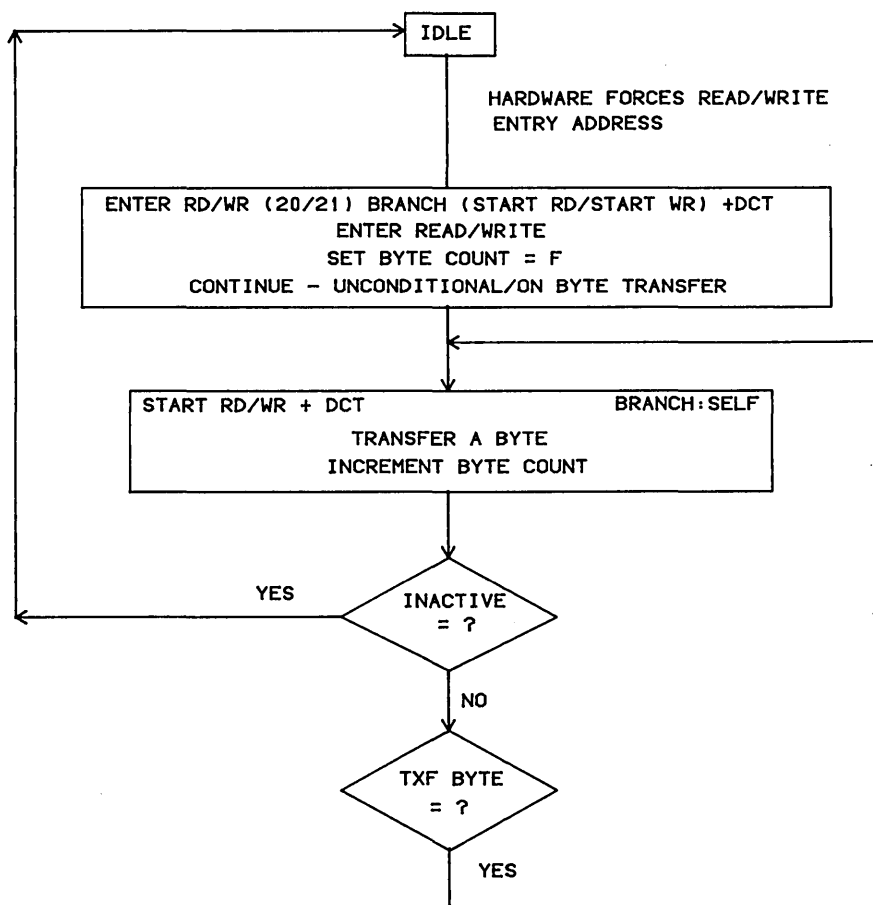
Full Word Transfers

The flow diagram in figure III-3-4 is valid for the following operations:

Read/Write - IOU BR, CM BR, FS1, MR, CM CEL, CM UEL1.
Read Only - IOU#SS, CPU#SS, CM#SS

The instructions are differentiated from one another by the selects generated in the last nanocrand. The last nanocrand loops on itself, providing wraparound read/write capability. The nanocode sequence is terminated (nanocode goes to idle address) on termination of the transfer by MCU.

The DCT is formed from a combination of the second control word (register specification within a mainframe element) and the type code (element specification).



C1047

Figure III-3-4. Full Word Transfers Flow Diagram

Partial Word Transfers

When a register is less than eight bytes in length, the valid bytes are not necessarily the highest or lowest order bytes in the register. All register reads/writes begin with the high order byte. When a byte is not valid, zeros are read and incoming data is ignored.

The flow diagram in figure III-3-5 is valid for the following operations:

- Read/Write - PFS, PTM, EC, FS2, TCM, CM CEL, CM UEL2
- Read Only - IOU Status, Option Installed, CSEL

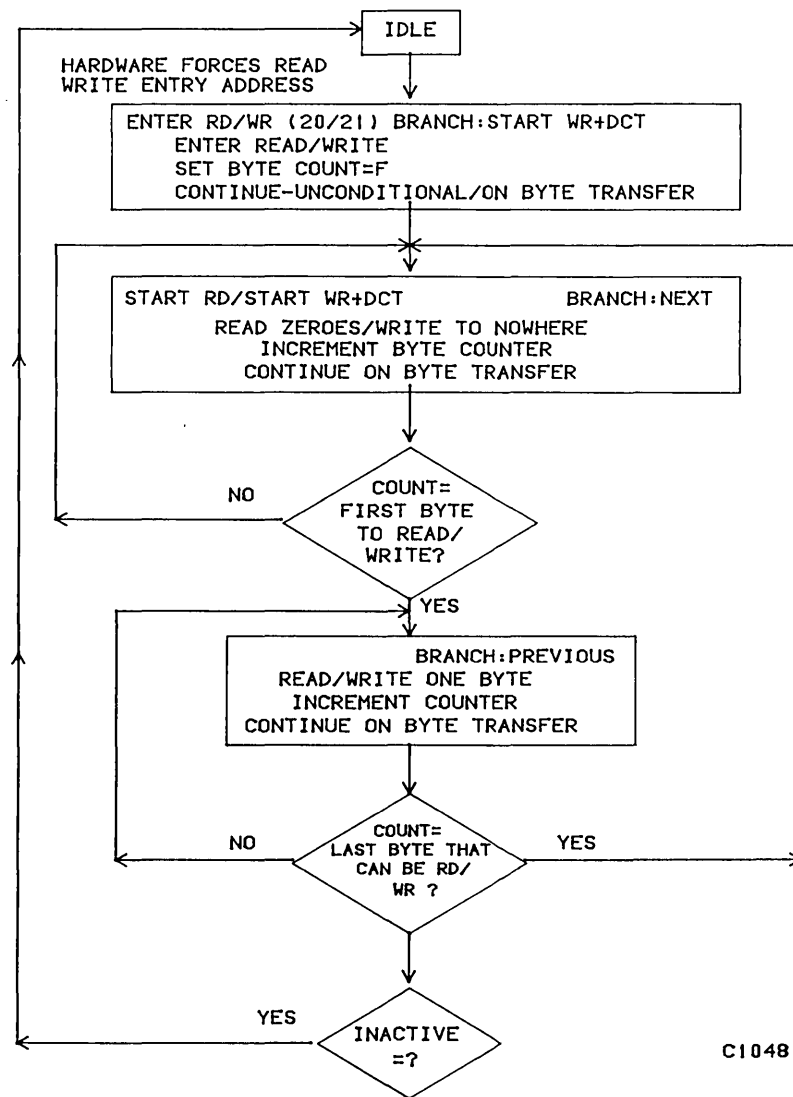


Figure III-3-5. Partial Word Transfers Flow Diagram

Some special cases occur where logical and physical byte numbers do not correspond. To access the bytes required, the byte counter must be reset. These are:

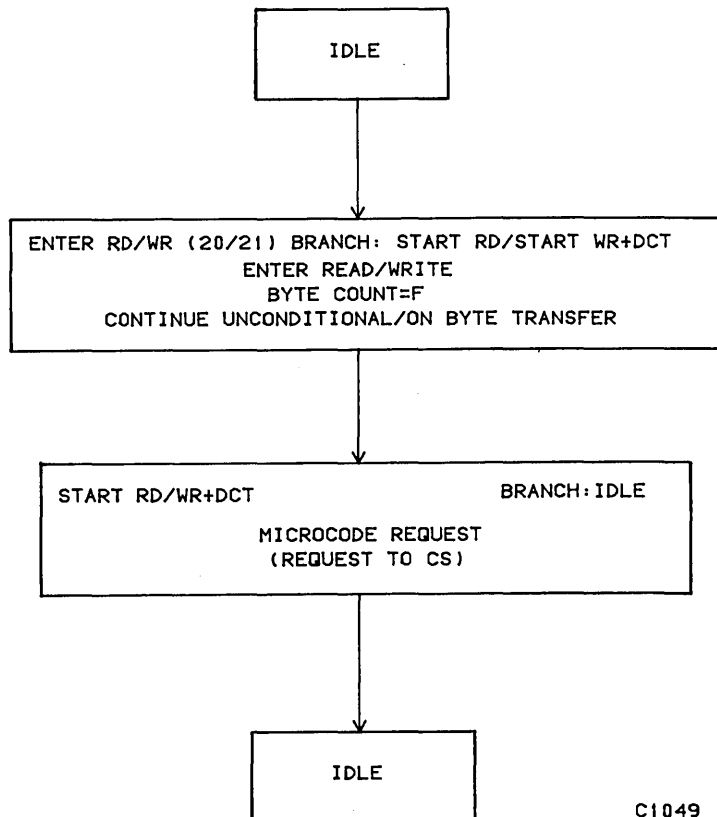
Read/Write - MAP CEL, S, BKPT
Read - PID

NOTE

To reset the byte counter for wraparound, more nanocrands are required in these cases.

MAC SEQUENCES--MICROCODE REQUIRED

When microcode is required to perform an operation, MAC passes a microcode request to the selected CPU and exits until that CPU requests a data transfer. Refer to figure III-3-6. To MAC, this appears no different than any other CPU request and no connection is made to the original MAC request.



C1049

Figure III-3-6. Microcode Request

An example would be reading the P register. When this instruction is entered in MAC, the microcode request and MAC select codes are sent, MAC accept is received, and MAC exits. The selected CPU moves the contents of the P register to the shared AD register and requests that the MAC move the contents of AD to the MCH. The RGC field of CPU microcode controls this operation; its format is as follows:

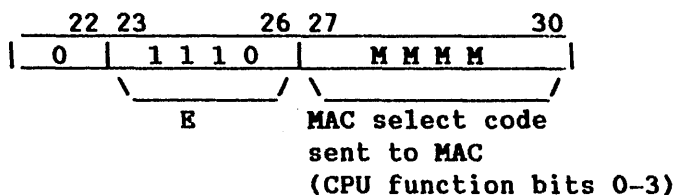
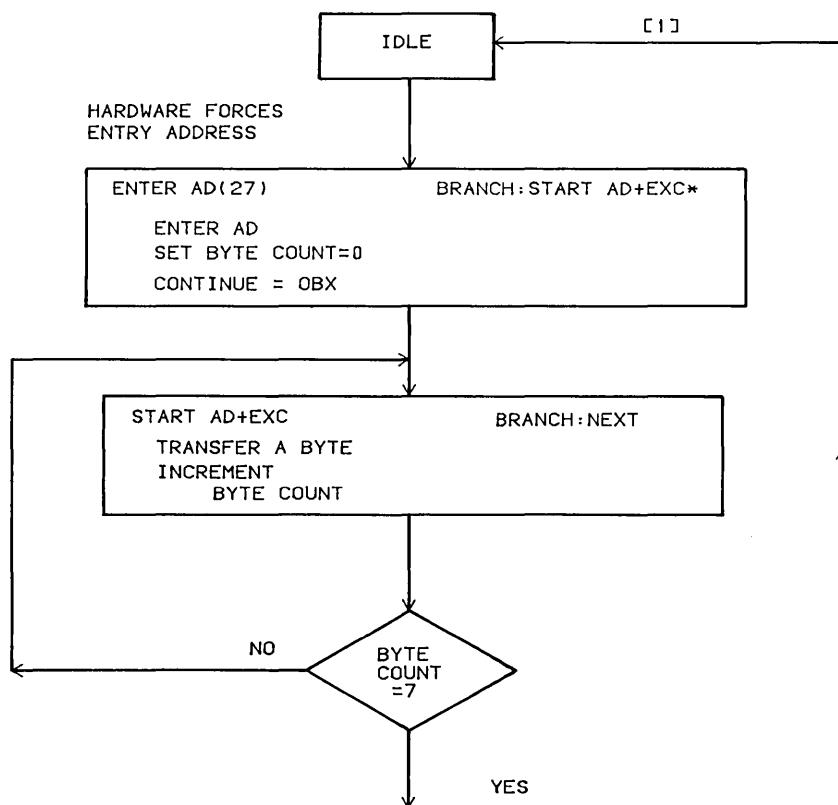


Figure III-3-7 shows MAC nanocode flow for all operations requiring a CPU register with a soft, execute, or map specification. If the processor is halted, a request cannot be sent; instead, MAC writes to nowhere or, in the case of a read, echoes back the second CW.



* MAC SELECT CODE

C1050

Figure III-3-7. Full Word Transfer--Microcode Required

CPU Processor Requests

These operations are specified by CPU through the use of a MAC select code (CPU function bits 0 through 3, MAC 2.3). This is used to enter the nanocode at the starting address of the corresponding nano-program. All these transfers involve the AD register (the CPU/MAC common register) and either MAC register or maintenance channel.

The MAC select code (CPU function bits 0 through 3) specifies the registers involved in these transfers as follows:

<u>Code</u>		<u>Register</u>
0 CW	→	AD
1 OI	→	AD
2 CEL	→	AD
3 PTM	→	AD
4 PFS	→	AD
5 PID	→	AD
6 EID	→	AD
7 MCH	→	AD
8		
9		
A AD	→	CEL
B AD	→	PTM
C AD	→	PFS
D		
E AD	→	KEYPOINT
F AD	→	MCH

From the point of view of the nanocode, these are the same as the transfers specified by MCU in which no microcode is required except there is a single entry point for all these operations (enter AD). The select code specifies the operation uniquely, and there is no wraparound read/write; a transfer consists of one CM word (eight bytes) after which the nanocode exits. The Read CW operation transfers only one byte to the CPU before exiting--the second CW sent by the MCU.

Full Word Transfers

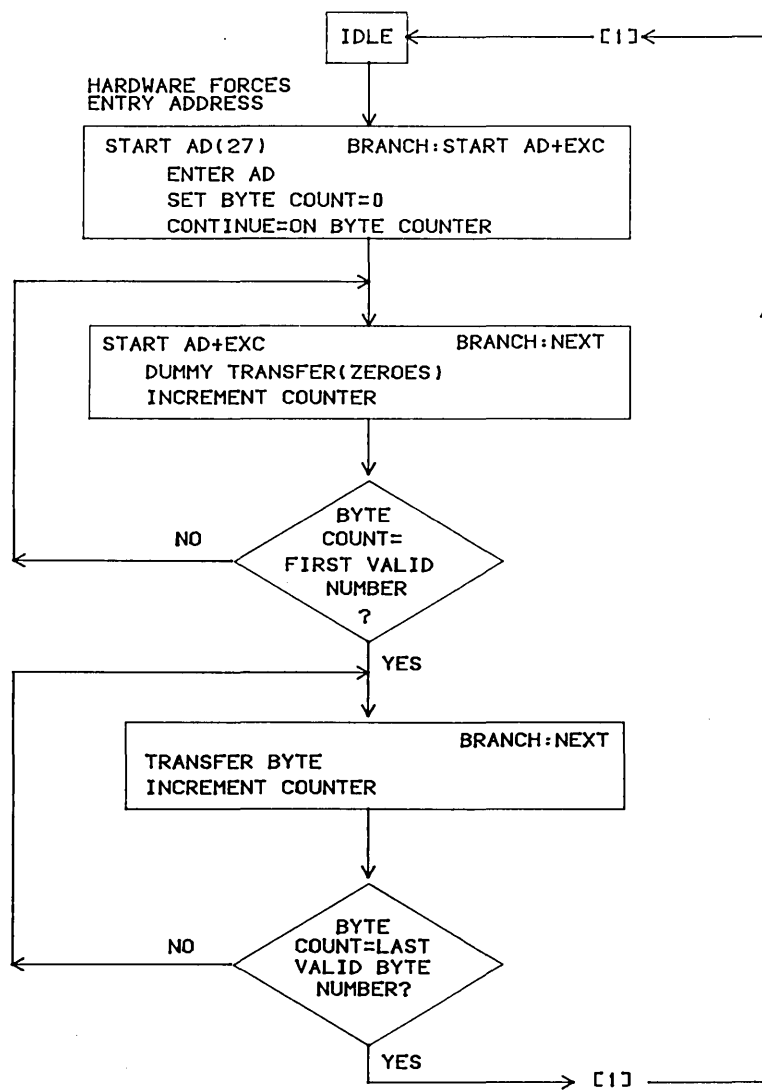
This is analagous to the instructions under MCU request, no microcode required, above, where a full 64-bit word of valid data is being transferred. The AD to channel instruction is performed as a MAC-accessible register read when CS is halted. Refer to figure III-3-7.

Instructions of this form are:

CH to AD
AD to CH

Partial Word Transfers

The flow diagram for this type of transfer is shown in figure III-3-8.



C1051

Figure III-3-8. Partial Word Transfer, Microcode Required

SECTION IV-1

INTRODUCTION

=====

The BS167-A Central Memory (CM) with appropriate memory increments can store 262K to 2096K 64-bit data and eight bit parity words. The BS222-A memory option can store 2096K to 4198K 64-bit data, 8-bit parity words.

Memory increments consist of additional memory array paks. Table IV-1-1 shows the paks to be installed for each of the memory sizes. A pak placement diagram in the System Multi-level Block Diagrams Manual shows installation positions for each equipment option.

TABLE IV-1-1. MEMORY SIZES AND EQUIPMENT REQUIREMENTS

Memory Size (72-Bit Words)	Equipment Requirements	Memory Array Paks per Equipment	
		Type	Number
262K	BS167-A	ODPH	2
524K	BS167-A	ODPH	4
1048K	BS167-A	ODPH	8
1572K	BS167-A	ODPH	12
2096K	BS167-A	ODPH	16
2096K ⁰⁰	BS222-A	3HBH	4
4198K	BS222-A	3HBH	8

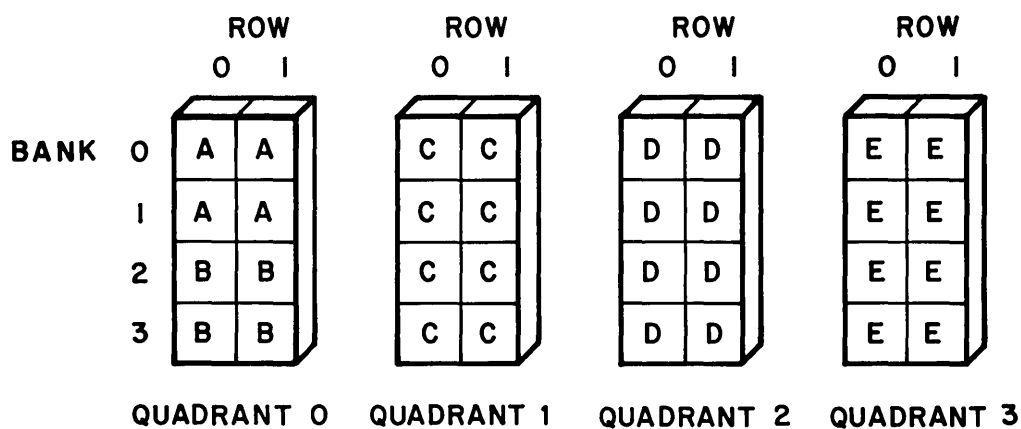
Logic paks and memory paks are identifiable by four-character designators (for example, 3HBH). For brevity, only the two middle letters (for example, HB) are mentioned in related text.

DESCRIPTION

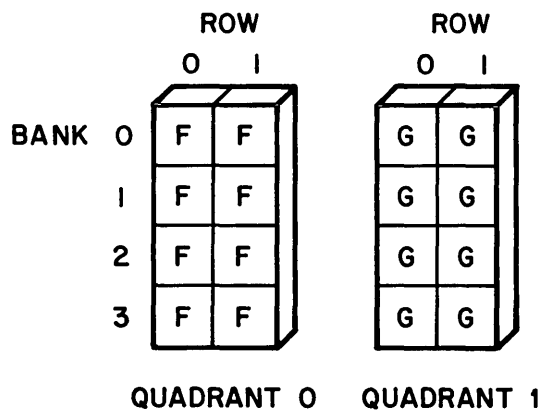
The CM is a word-addressable random access memory (RAM) contained in two or four memory banks (figure IV-1-1).

Memory banks store 2, 4, 8, 12, 16, or 32 megabytes with a 72-bit word (64 data and 8 parity or ECC bits) and have logical divisions consisting of two rows. In a minimum memory configuration of 262K words, the memory contains rows 0 and 1 and banks 0 and 1 in quadrant 0. An additional 262K-word increment adds banks 2 and 3 in the same quadrant. Maximum configuration has four banks arranged in four quadrants.

Single error correction/double error detection (SECCDED) generators generate ECC bits stored with each word. SECCDED generates SYNDROME bits, checks circuits, corrects single-bit errors, and reports double-bit errors.



$4A = 2 \text{ MB} = 262 \text{ K WORDS}$
 $4A + 4B = 4 \text{ MB} = 524 \text{ K WORDS}$
 $4A + 4B + 8C = 8 \text{ MB} = 1048 \text{ K WORDS}$
 $4A + 4B + 8C + 8D = 12 \text{ MB} = 1572 \text{ K WORDS}$
 $4A + 4B + 8C + 8D + 8E = 16 \text{ MB} = 2096 \text{ K WORDS}$
 ONE QUADRANT = 4 ODPH (64 K CHIPS) PAKS



$8F = 16 \text{ MB} = 2096 \text{ K WORDS}$
 $8F + 8G = 32 \text{ MB} = 4192 \text{ K WORDS}$
 ONE QUADRANT = 4 3HBH (256 K CHIPS) PAKS

C1060

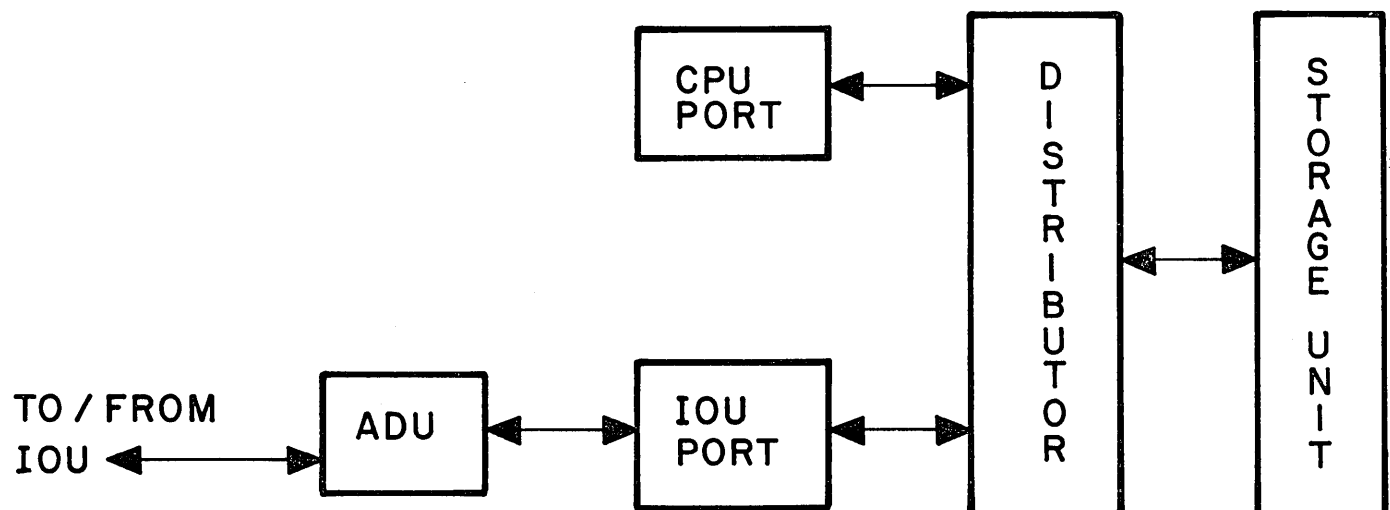
Figure IV-1-1. CM Organization

Circuits in the CM use 10K emitter-coupled logic (ECL) on all logic and memory array paks, plus one ECL macrocell chip on the control pak (DC). The memory array paks also use the 256K or 64K x 1 MOS dynamic RAM with TTL levels. ECL 10K translators are used to convert between ECL and TTL.

The logic voltage levels are shown below:

Logic State	Voltage Level	
	ECL	TTL
Low	-1.75V	0V
Hi	-0.90V	+5V

The CM operates on a 50 ns clock. It consists of four major elements (assembly/disassembly unit, ports, distributor, and storage unit) as shown in figure IV-1-2.



CI06I

Figure IV-1-2. Central Memory Block Diagram

ASSEMBLY/DISASSEMBLY UNIT (ADU)

Unit Description

The assembly/disassembly unit (ADU) provides the interface between CM and IOU (figure IV-1-2). It controls, assembles, and disassembles the data and checks the data flow between CM and IOU. The ADU operates in two modes to accommodate the two CM formats: 60-bit mode and 64-bit mode. A 60-bit CM word is assembled from five 12-bit PP words and disassembled into five 12-bit PP words. A 64-bit word is assembled from four 16-bit PP words and disassembled into four 16-bit PP words. Each word is read/written from/to successive locations in PPM.

ADU Priority

IOU MR EC register bits 56 and 57 are used in controlling the ADU priority which allows software to select either BLOCK mode priority or WORD mode priority for each barrel. The mode can be changed while the IOU is running. These two bits may be set to select any one of the four possible priority modes. Priority switch from one PP to the partner PP in the opposite barrel requires one major cycle.

A PP running in BLOCK mode releases priority for access to the ADU in CM at the end of the CM instruction. In WORD mode, the PP releases at the end of each CM word transferred if another PP is requesting ADU priority at the same time.

PORTS

The two CM ports, CPU and IOU, make it accessible to the CP and every PP. The IOU port is connected to the IOU, while the CPU port is connected to the CPU. In a dual CPU configuration, both CPUs share the one port. The port is made available to each CPU on an alternating basis (every other clock period).

DISTRIBUTOR

The distributor resolves port conflicts and multiplexes data from ports to the storage unit. There is no long term lockout of any port. It contains the error correction code (ECC) generator, SECDED, and partial write logic. An ECC is generated prior to sending the data word to the storage unit; SECDED logic checks data read from the storage unit.

STORAGE UNIT

The storage unit consists of memory arrays with data, address, and control logic interfaces. With 64K-chip arrays, it is expandable from 2 to 4, 8, 12 and 16 megabytes. With 256K-chip arrays, it is expandable to 16 or 32 megabytes. The storage unit has two banks for two megabytes and four banks for other memory sizes. It is a dynamic random access memory with a word length of 72 bits (64 data and eight parity or code bits).

CM FUNCTIONS

The CM performs the following functions:

- Read
- Write
- Read and set lock
- Read and clear lock
- Exchange
- CYBER 170 exchange request
- Read exchange address
- Read free running counter
- Refresh counter resync
- Interrupt

Refer to Memory Operations in section 4 of this part for detailed description of each function.

CM FEATURES

The CM has the following features:

- Parity
- Maintenance registers
- SECDED
- Bounds register
- Memory configuration switch register

Refer to CM Reliability, Accessibility, and Maintainability Features in section 5 of this part for detailed description of each feature.

SECTION IV-2

ASSEMBLY DISASSEMBLY UNIT

=====

The assembly/disassembly unit (ADU) provides the interface between CM and IOU. It controls, assembles, and disassembles data, and also checks the data flow between CM and IOU. This section describes the ADU instruction format, configuration, addressing, data transfer/manipulation, and its major circuit components.

ADU INSTRUCTION

The following ADU instructions transfer data to/from the CM and IOU. For more information refer to appendix B.

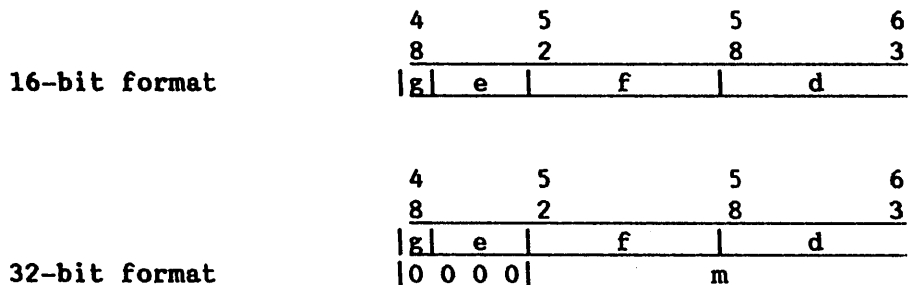
<u>Code</u>	<u>Description</u>
0060d	Central read from ((R)+(A)) to d
1060d	Central read from ((R)+(A)) to d
0061dm	Central read (d) words from ((R)+(A)) to m
1061dm	Central read (d) words from ((R)+(A)) to m
0062d	Central write to ((R)+(A)) from d
1062d	Central write to ((R)+(A)) from d
0063dm	Central write (d) words to ((R)+(A)) from m
1063dm	Central write (d) words to ((R)+(A)) from m

The following ADU instructions load/store data to/from the PPM and R register. For more information refer to appendix B.

<u>Code</u>	<u>Description</u>
0024d	Load R register
0025d	Store R register

ADU INSTRUCTION FORMAT

All ADU instructions are represented in one of two formats. One uses a single 16-bit word; the other uses two consecutive 16-bit words.



The following field descriptions apply to both instruction formats.

g	1 bit	Most significant bit of 7-bit operation code. In these instructions, g controls the width of operand read from PPM. If g is 0, operand is 12 bits, if g is 1, operand is 16 bits.
e	3 bits	Not used, must be zero.
f	6 bits	Least significant six bits of 7-bit operation code.
d	6 bits	Word count or address specification depending on instruction.
m	12 bits	PPM address specification.

CONFIGURATION

There is one ADU and an associated IOU port for controls and data transfer.

ADU handles both 60- and 64-bit words. The 60-bit word is assembled/disassembled from/to five 12-bit PPM words. The four unused bits in the 60-bit word are zero filled prior to writing to CM. The 64-bit word is assembled/disassembled from/to four 16-bit PPM words.

The 28-bit CM address (of which 23 bits are sent to CM) is formed either directly using the contents of the A register or by address relocation using the contents of the R register and A register as described below.

CM ADDRESSING

The PPs have access to all CM storage locations. The CM address is formed from the contents of the R register (BAS 2.2) and the A register (BAS 2.0). If bit 46 of the A register is zero, then direct addressing is performed and bits 47 to 63 of A register make up the CM address (BAS 2.3). The R register contains an absolute 64-word starting boundary within CM. If bit 46 of the A register is set, then address relocation is performed and the CM address is formed as follows: bits 47 to 57 of A register are added to the 22-bit R register (bits 36-57) in the R+A adder (BAS 2.3) and then bits 58 to 63 of A register are concatenated to the rightmost end of the 22-bit result from the R+A adder to form the 28 bit CM address. Only 23 bits (bits 41-63) are sent to CM. The left-most five bits (bits 36 to 40) are not sent to CM and together with bits 41 to 53, they are used for address out of bounds checking (BAS 2.3). Bit 46 of A register is not included in the sum.

The address in A register rank 9 (BAS 2.0) feeds the R+A adder (BAS 2.3), and CM address register rank 0 (BAS 2.3). A register bit 46 selects either R+A or A as shown in table IV-2-1.

The CM address rank 9 bits 36 to 57 input CM address rank 0 register (BAS 2.3) and then feed RA mux. CM address rank 0 bits 36 to 53 are checked against OSB data bits for OSB error (BAS 2.3) as shown below.

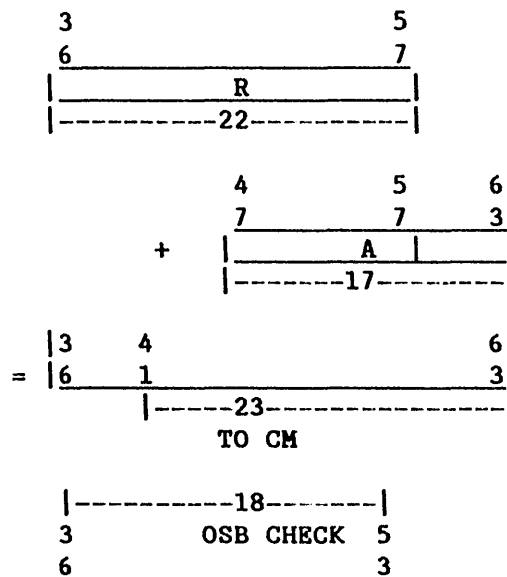


TABLE IV-2-1. CM ADDRESS FORMATION

A Rgtr Bit 46	CM ADDRESS BITS			Function
	36-46	47-57	58-63	
0	0	A Rgtr 47-57	A Rgtr 58-63	Direct Address
1	R Rgtr 36-46 + Carry over from bit 47 of R+A	R+A 47-57	A Rgtr 58-63	Adrs Relocation

ADU DATA TRANSFERS/MANIPULATIONS

60-BIT DISASSEMBLY (CM TO PPM)

Central Read from (A) to d (0060 d)

Bits 4 to 63 of one CM word are transferred to bits 52 to 63 of five consecutive PPM words. Bits 0 to 3 of the CM word are discarded and the remaining 60 bits are disassembled from the left into five 12-bit words. This unpacking is illustrated below (ADU 2.1).

One parity bit is stored in ADU for each four data bits. The disassembly mux (ADU 2.2) selects 16 data bits and four parity bits (each DD pak selects four data bits and one parity bit). New parity is generated from the four parity bits associated with the four DD paks. The 16 data bits and the new parity bit pass through CM read data register (ADU 2.2) to be disassembled by the

Y mux (PPM 2.2). The leftmost four data bits and the parity bit are discarded. PPM data in regenerator/checker generates one parity bit for the remaining 12 bits (PPM 2.2).

Central Memory Word Within ADU

0	4	1	2	4	5	6
6	8	0	2	3		
(4)	a(12/3)	b(12/3)	c(12/3)	d(12/3)	e(12/3)	

PPM Words

	4	5	6
	8	2	3
d	0(4)	a(12/1)	
d+1	0(4)	b(12/1)	
d+2	0(4)	c(12/1)	
d+3	0(4)	d(12/1)	
d+4	0(4)	e(12/1)	

The address of the first PPM word is specified by d. This address is written into the Q register (BAS 2.4). The Q register increments the PPM address by one every time a 12-bit word is transferred to PPM from CM.

Central Read (d) Words from (A) to m (0061 d m)

Central read (d) words from (A) to m transfers bits 4 to 63 of consecutive CM words to bits 52 to 63 of consecutive PPM words. Bits 0 to 4 of each CM word are discarded and the remaining 60 bits are disassembled from the left into five 12-bit words. See 060 instruction for illustration of unpacking.

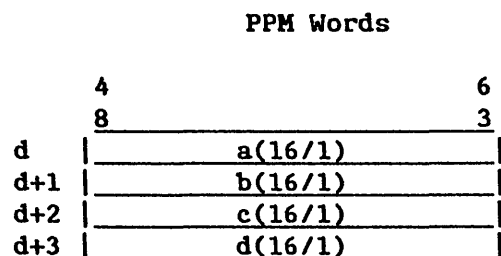
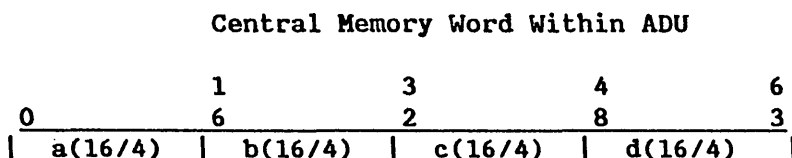
The address of the first PP word is specified by m. This address is written into the P register (BAS 2.5) while the program address is written in PPM location 0000. The P register increments the PPM address by one every time a 12-bit word is transferred from CM to PPM. The number of CM words is specified by (d). This value is written into the Q register (BAS 2.4). Every time a CM word is transferred to PPM, Q register decrements by one.

If the value of A register (BAS 2.0) exceeds 377777, then bit 46 of A register is set. This causes the operation to switch between direct address and relocation address modes. If the last word transferred is from a relative address of 777776, then upon completion, the A register is zero and does not point to the address plus one of the last word transferred.

64-BIT DISASSEMBLY (CM TO PPM)

Central Read from (A) to d Long (1060 d)

One CM word is transferred to bits 48 to 63 of four consecutive PPM words. The CM word is disassembled from the left into four 16-bit words. This unpacking is illustrated below (ADU 2.1).



The address of the first PPM word is specified by d. This address is written into the Q register (BAS 2.4). The Q register increments the PPM address by one every time a 16-bit word is transferred to PPM from CM.

Central Read (d) Words from (A) to m Long (1061 d m)

Central read (d) words from (A) to m long transfers consecutive CM words to consecutive PPM words. Each CM word is disassembled from the left. See the 1060 instruction for illustration of this unpacking.

The address of the first PPM word is specified by m. This address is written into the P register (BAS 2.5), while the program address is stored in PPM location 0000. The P register increments the PPM address by one every time a 16-bit byte is transferred from CM to PPM. The number of CM words transferred is specified by (d). This value is written into the Q register (BAS 2.4). Every time a CM word is transferred to PPM Q register decrements by one.

If the value of A register (BAS 2.0) exceeds 377777, then bit 46 of A register is set. This causes the operation to switch between direct address and relocation address modes. If the last word transferred is from a relative address of 777776, then upon completion, the A register is zero and does not point to the address plus one of the last word transferred.

60-BIT ASSEMBLY (PPM TO CM)

Central Write from d to (A) (0062 d)

Bits 52 to 63 of five consecutive PPM words are transferred (bits 48 to 51 of the words are ignored) to bits 4 to 63 of one CM word (bits 0-3 are cleared). These bytes are assembled from the left as illustrated below (ADU 2.1).

PPM Words			
	4	5	6
	8	2	3
d	(4)	a(12/3)	
d+1	(4)	b(12/3)	
d+2	(4)	c(12/3)	
d+3	(4)	d(12/3)	
d+4	(4)	e(12/3)	

Central Memory Word Within ADU

0	4	1 6	2 8	4 0	5 2	6 3
(4)	a(12/3)	b(12/3)	c(12/3)	d(12/3)	e(12/3)	

The address of the first PP word is specified by d. This address is written into the Q register (BAS 2.4). The Q register increments the PPM address by one every time a 12-bit byte is transferred from PPM.

Central Write (d) Words from m to (A) (0063 d m)

Central write (d) words from m to (A) transfers bits 52 to 63 of consecutive PPM words to bits 4 to 63 of consecutive CM words. See the 0062 instruction for illustration of this packing.

The address of the first PPM word is specified by m. This address is written into the P register (BAS 2.5), while the program address is stored in PPM location 0000. The P register increments the PPM address by one every time a 12-bit byte is transferred from PPM. The number of CM words transferred is specified by (d). This value is written into the Q register (BAS 2.4). Every time a CM word is transferred from PPM to CM Q register decrements by one.

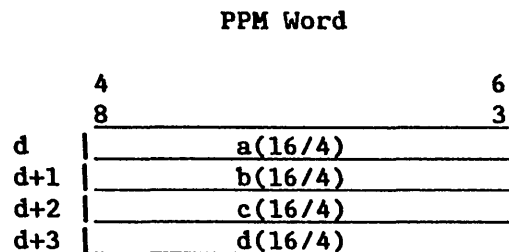
If the value of A register (BAS 2.0) exceeds 377777, then bit 46 of A register is set. This causes the operation to switch between direct address and relocation address modes. If the last word transferred is from a relative address of 777776 then, upon completion, A register is zero and does not point to the address plus one of the last word transferred.

64-BIT ASSEMBLY (PPM TO CM)

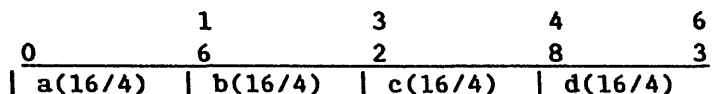
Central Write from d to (A) Long (1062 d)

Four consecutive PPM words are transferred to one CM word. This packing is illustrated below (ADU 2.1).

The address of the first PPM word is specified by d. This address is written into the Q register. The Q register increments the PPM address by one every time a 16-bit byte is transferred from PPM.



Central Memory Word Within ADU



Central Write (d) Words from m to (A) Long (1063 d m)

Central write (d) words from m to (A) long transfers consecutive PPM words to consecutive CM words. Four PPM words are packed from the left into each CM word.

The address of the first PPM word is specified by m. This address is written into the P register (BAS 2.5), while the program address is stored in PPM location 0000. The P register increments the PPM address by one every time a 16-bit byte is transferred from PPM. The number of CM words transferred is specified by (d). This value is written into the Q register (BAS 2.4). Every time a CM word is transferred from PPM the Q register decrements by one.

If the value of A register (BAS 2.0) exceeds 377777, then bit 46 of A register is set. This causes the operation to switch between direct address and relocation address modes. If the last word transferred is from a relative address of 777776 then, upon completion, the A register is 0 and does not point to the address plus one of the last word transferred.

12-BIT/16-BIT WORD

The ADU operates with either a 12-bit or 16-bit PPM word. Micrand bit 22 from microcode ROM (BAS 2.8) selects the 16-bit PPM word.

ADU MAJOR CIRCUIT COMPONENTS

ADU ASSEMBLY MUX

The ADU assembly mux (ADU 2.0) controls and selects 12/16 bit PPM words for assembly into a 60/64 bit CM word. The ADU PPM nibble parity mux (ADU 2.1) controls and selects 3/4 parity bits for 12/16 bit PPM words. Both muxes are controlled by Micrand Rank 8 bits 73 to 75 (word counts 0 to 2) and Micrand Rank 8 bit 22 (Enable 16-bit Mode) (BAS 2.8).

For 16 bit PPM words assembling into one 64 bit CM word, there are four parcels and each parcel is 16 bits wide (ADU 2.0). The parities associated with each parcel are the parities generated by PPM nibble parity generator P0-P3 (ADU 2.1).

For 12-bit PPM words assembling into one 60-bit CM word, there are five parcels and each parcel is 12 bits wide. First parcel is PPM Data Rank 9 bits 52 to 63 which pass straight through the assembly mux. Second parcel is PPM Data Rank 9 bits 52 to 63 left-shifted four places. Third parcel is PPM Data Rank 9 bits 52 to 63 left-shifted eight places. Fourth parcel is PPM Data Rank 9 bits 52 to 63 left-shifted twelve places. Fifth parcel is PPM Data Rank 9 bits 52 to 63 straight through the assembly mux again.

PPM nibble parity mux selects the three parities associated with each of the five 12-bit parcels. Table IV-2-2 shows assembly/disassembly mechanism for both data and parity.

ADU READ/WRITE BUFFER

ADU read/write buffer is a 16 x 64-bit RAM that stores a 64-bit CM word during assembly/disassembly (ADU 2.2). Each DD pak has four RAM chips for data and four RAM chips for parity for each barrel.

Pak Address bits 62 and 63 (ADU 2.3) select one of the four chips on the pak and Pak Write Enable (ADU 2.3) selects that particular pak. Table IV-2-3 shows the assignment of Pak Address bits 62 and 63 and Pak Write Enable for each parcel for both 12-bit and 16-bit modes.

DATA TO ADU DISASSEMBLY MUX

ADU disassembly is done in two stages. First stage disassembly is at data to ADU disassembly mux (ADU 2.2). Pak Address bits 62 and 63 select one of the four RAM chip group data bits. These four ADU data bits from each DD pak clock through CM read register (ADU 2.2). Second stage disassembly is at Y mux (PPM 2.2) where 16 ADU data bits (from four DD paks) are further disassembled.

TABLE IV-2-2. 12/16 BIT ASSEMBLY/DISASSEMBLY

CM WORD WITHIN ADU

BYTE BIT	0L 0-3	0R 4-7	1L 8-11	1R 12-15	2L 16-19	2R 20-23	3L 24-27	3R 28-31	4L 32-35	4R 36-39	5L 40-43	5R 44-47	6L 48-51	6R 52-55	7L 56-59	7R 60-63
16 BIT PPM WORD	48 - 51	52 - 55	56 - 59	60 - 63	48 - 51	52 - 55	56 - 59	60 - 63	48 - 51	52 - 55	56 - 59	60 - 63	48 - 51	52 - 55	56 - 59	60 - 63
12 BIT PPM WORD		52 - 55	56 - 59	60 - 63	52 - 55	56 - 59	60 - 63	52 - 55	56 - 59	60 - 63	52 - 55	56 - 59	60 - 63	52 - 55	56 - 59	60 - 63
DD PAK NIBBLE	1 P0	2 P0	3 P0	4 P0	1 P1	2 P1	3 P1	4 P1	1 P2	2 P2	3 P2	4 P2	1 P3	2 P3	3 P3	4 P3
WRITE (ASSEMBLY)	STRAIGHT				LEFT 4				LEFT 8				LEFT 12			
PAK WRITE ENBL	1 2 3 4				1 2 3				4 1 2				3 4 1			
READ (DISASSEMBLY)	STRAIGHT				RIGHT 4				RIGHT 8				RIGHT 12			

C1062

NIBBLES STROBES FOR 60/64 BIT WRITE

12 BIT PP WORD	DD PAK 1				DD PAK 2				DD PAK 3				DD PAK 4			
	P0	P1	P2	P3	P0	P1	P2	P3	P0	P1	P2	P3	P0	P1	P2	P3
1	●				●				●				●			
2		●				●				●				●		
3			●				●				●				●	
4				●				●				●				●
5									●							
16 BIT PP WORD	DD PAK 1				DD PAK 2				DD PAK 3				DD PAK 4			
	P0	P1	P2	P3	P0	P1	P2	P3	P0	P1	P2	P3	P0	P1	P2	P3
1	●				●				●				●			
2		●				●				●				●		
3			●				●				●				●	
4				●				●				●				●

C1063

TABLE IV-2-3. PAK ADDRESS BITS AND PAK WRITE ENABLE ASSIGNMENT

		PAK Address Bit 62,63				PAK Write Enable			
		1	2	3	4	1	2	3	4
16 Bit Mode	First Parcel	00	00	00	00	1	1	1	1
	Second Parcel	01	01	01	01	1	1	1	1
	Third Parcel	10	10	10	10	1	1	1	1
	Fourth Parcel	11	11	11	11	1	1	1	1
12 Bit Mode	First Parcel	00	00	00	00	1	1	1	1
	Second Parcel	01	01	01	XX	1	1	1	0
	Third Parcel	10	10	XX	01	1	1	0	1
	Fourth Parcel	11	XX	10	10	1	0	1	1
	Fifth Parcel	XX	11	11	11	0	1	1	1

XX is don't care.

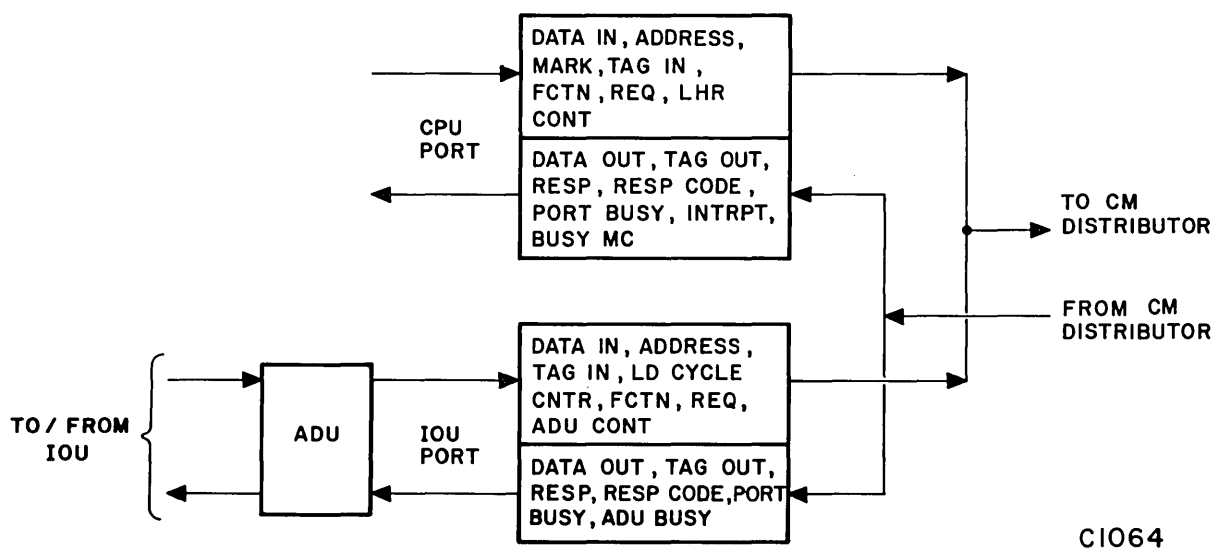
For 16-bit mode disassembly, four 16-bit PP words are formed. For 12-bit mode disassembly (the reverse of 12-bit mode assembly), five 12-bit PP words are formed (table IV-2-1). First parcel is ADU Data bits 52 to 63 straight through Y mux (PPM 2.2). Second parcel is ADU Data bits 48 to 59 (ADU Data bits 48 to 63 right-shifted four places). Third parcel is ADU Data bits 60 to 63 and 48 to 55 (ADU Data bits 48 to 63 right-shifted eight places). Fourth parcel is ADU Data bits 56 to 63 and 48 to 51 (ADU Data bits 48 to 63 right-shifted 12 places). Fifth parcel is ADU Data bits 52 to 63 straight through Y mux again.

Besides its disassembly functions, Y mux also selects A bypass rank 0 data, MAC/RA mux data, channel data and P rank 0 data to PPM.

SECTION IV-3

PORT INTERFACES

CM has two ports: IOU and CPU port (figure IV-3-1). CPU port communicates with CPU in a single CPU configuration. In a dual CPU configuration, both CPUs share the one port. Maximum transfer rate is one data word every clock period. IOU port communicates with IOU through ADU. Maximum transfer rate is one word every two clock periods. Every input/output port signal for each port is described below.

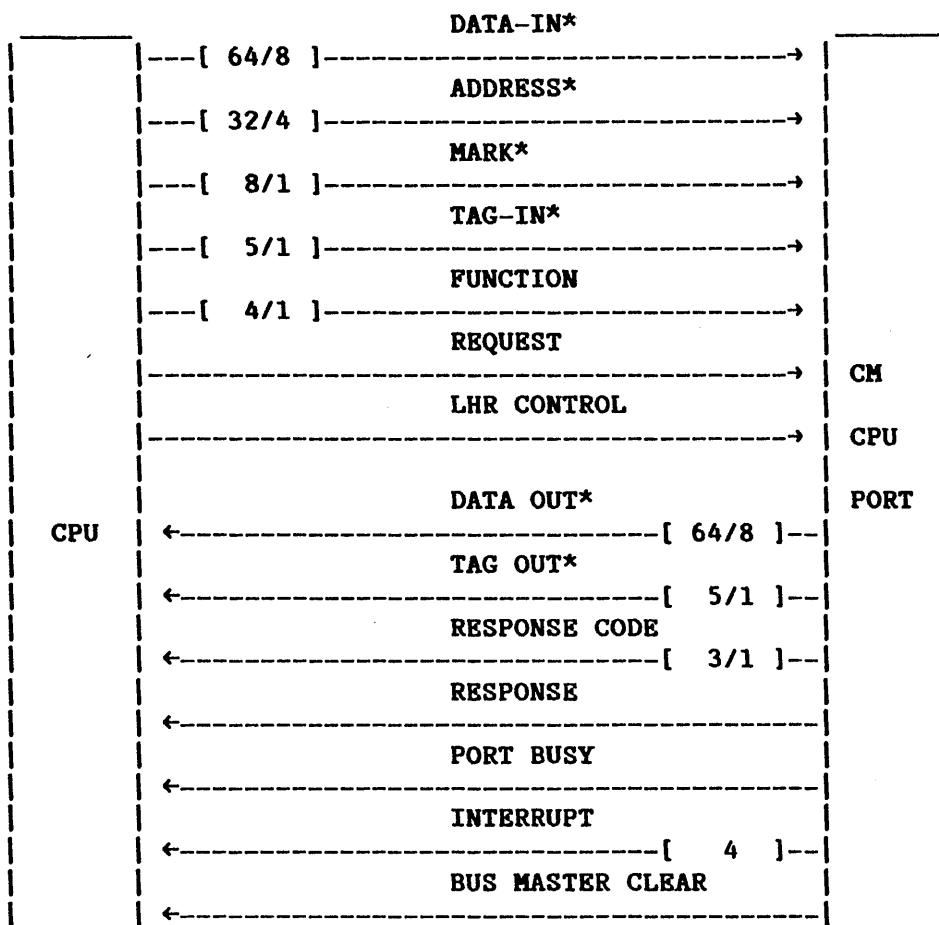


Off-sheet references are to CM Distributor, figure IV-4-1.

Figure IV-3-1. CM Ports

CPU/CPU PORT INTERFACE

This port interface to the processor(s) consists of 130 input lines (123 of which are shared in a dual CPU configuration) and 89 output lines (78 of which are shared in a dual CPU configuration). The parity lines carry odd parity. Refer to figure IV-3-2.



(*) Shared by both CPUs in a dual CPU configuration.

Figure IV-3-2. CPU/CPU Port Interface

PORT INPUT LINES

*Data-in Lines (64 data and 8 parity) carry information to be stored in memory during write operations (MEM 2.2). A parity bit accompanies each byte of data. The content of the data-in lines on a nonwrite operation is undefined but has correct parity. The data-in lines also indicate which CPU is selected during an interrupt or exchange operation.

*Address Lines (32 address and 4 parity) specify address bits 32 to 63 with four parity bits (MEM 2.0). The content of the address lines on refresh counter resync or interrupt operations is undefined but has correct parity. Refer to section 5 of this part for addressing scheme.

*Mark Lines (8 mark and 1 parity) indicate which bytes are valid (bit 0 indicates byte 0, bit 1 indicates byte 1, etc.) within the data-in word during partial write operation. One parity bit accompanies the mark lines. The content of the mark lines on read type operations is undefined but has correct parity (MEM 2.4). Refer to Memory Operations for details.

*Tag-in Lines (5 tag and 1 parity) contain identifying information for the requesting processor during a request. This tag information is returned unmodified to the requesting processor through tag-out lines. One parity bit accompanies the tag lines (MEM 2.3). Refer to RNI 2.3.

Function Lines (4 function code and 1 parity) contain the desired function code for a given request. These four lines specify up to 16 functions of which ten are used. One parity bit accompanies the function lines (MEM 2.5). Refer to Memory Functions in section 4 of this part.

Request (1 line) strobes (Enable Strobe CPU Port on MEM 2.7) all port input signals such as address, mark, function code, and write data to the distributor.

LHR Control (1 line, CPU Fault on MEM 2.5), when set, turns the previous requested function to a read function. It sets when CPU has detected a fault associated with the CM request. This signal assures that no CM write will take place.

PORT OUTPUT LINES

*Data-Out Lines (64 data and 8 parity) contain the information being returned to the processor in response to a read (MEM 2.1). A parity bit accompanies each byte of data. On write type operations, contents of the data-out lines are zeros with correct parity.

*Tag-Out Lines (5 tag and 1 parity) contain a copy of the information placed on the tag-in lines during a request (MEM 2.3). The tag is returned to the processor exactly as received, one clock period prior to the corresponding data-out and response code lines.

Response Code Lines (3 response code lines and 1 parity) specify the nature of the response being returned to the processor. Three lines specify up to eight response codes. One parity bit accompanies the response code lines (MEM 2.3). Refer to Memory Responses in section 4 of this part.

Response (1 line) signal is sent to the processor with the tag-out lines and occurs one clock period prior to the corresponding data on the data-out and response code lines (MEM 2.6).

* Shared by both CPUs in a dual CPU configuration.

Port Busy (1 line) is sent to the CPU when the port buffer is full. It is a false signal and suspends CPU operation until the port becomes available (MEM 2.7).

Interrupt (4 lines). One is used for CYBER 180 interrupts, the other three lines specify the type of interrupt and are used for CYBER 170 exchange interrupts (MEM 2.5). Refer to Memory Operations for details.

Bus Master Clear (1 line) synchronizes the bus enable signals so that the CPUs alternate on the buses (MEM 2.7). It is derived from CM master clear.

IOU/IOU ADU PORT INTERFACE

This interface to the IOU consists of 68 input lines and 30 output lines. The parity lines carry odd parity. Refer to figure IV-3-3.

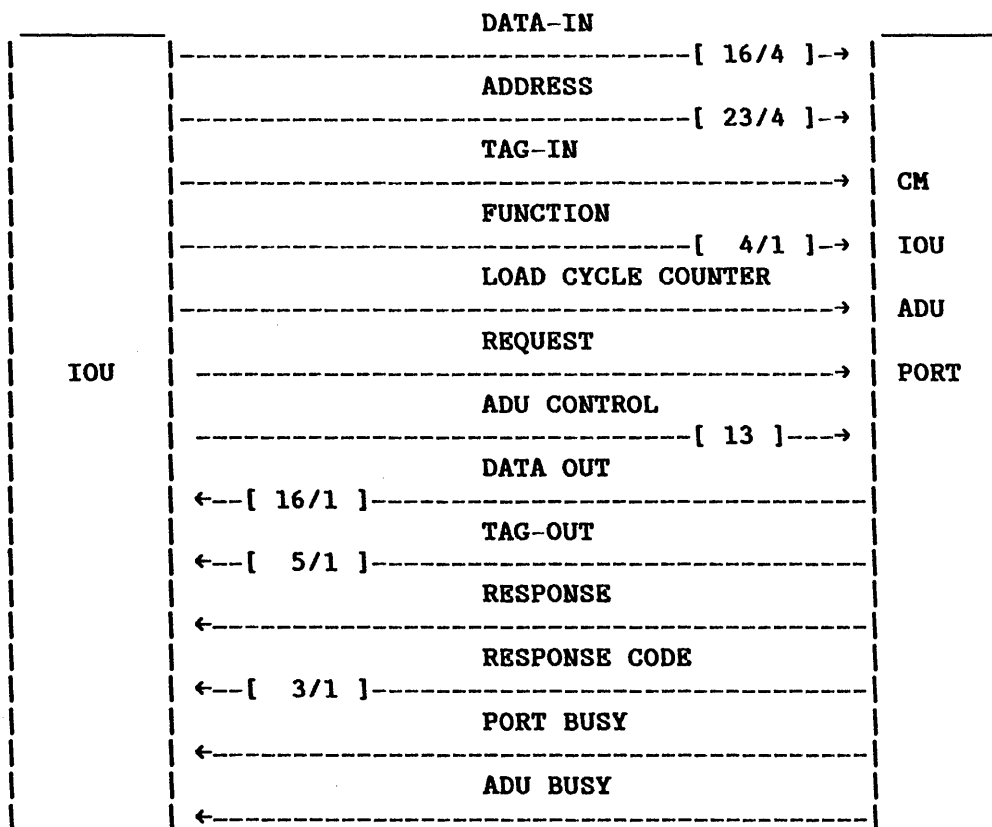


Figure IV-3-3. IOU/IOU ADU Port Interface

PORT INPUT LINES

Data-in Lines (16 data and 4 parity) contain the information to be assembled by the ADU to form a 64-bit data word to be stored in memory during write operations (ADU 2.0-2.2). The content of the data-in lines on a nonwrite operation is undefined but has correct parity.

Address Lines (23 address and 4 parity) specify address bits 38 to 60 with four parity bits (MEM 2.0). The content of the address lines on refresh counter resync or interrupt operations is undefined but has correct parity. Refer to section 5 of this part for addressing scheme.

Tag-in (1 line) indicates which barrel makes the request (logic zero indicates barrel 0; logic one indicates barrel 1). Circuitry synchronized with the IOU major cycle counter in the CMC generates the remaining tag bits which indicate the PP number (MEM 2.3).

Function Lines (4 function code and 1 parity) contain the desired function code for a given request. These four lines specify up to 16 functions of which ten are used. One parity bit accompanies the function lines (Micrand Rank 9 bits 79 to 83 on MEM 2.5).

Load Cycle Counter (1 line) synchronizes the tag circuitry with the IOU major cycle counter. It is a one clock period true signal (MEM 2.3).

Request (1 line) strobes all signals coming into the port. It is a false signal (Enable Strobe IOU Port on MEM 2.8).

ADU Control Lines (13 lines) control the operation of the ADU as it assembles and disassembles data. They consist of two pak address bits (ADU 2.2) and one pak write enable for each data pak (ADU 2.2), plus one 16-bit mode signal (ADU 2.3).

PORT OUTPUT LINES

Data-Out Lines (16 data and 1 parity) from ADU contain the information being returned to the processor in response to a read operation (ADU 2.0).

Tag-Out Lines (5 tag and 1 parity) contain a copy of the information placed on the tag-in lines and the PP number from the cycle counter during a request. The tag is returned to the processor one clock period prior to the corresponding data-out and response code lines (MEM 2.5).

Response (1 line) is sent to the processor with the tag-out lines and occurs one clock period prior to the corresponding data on the data-out and response code lines (MEM 2.6).

Response Code Lines (3 response code lines and 1 parity) specify the nature of the response being returned to the processor. Three lines specify up to eight response codes. One parity bit accompanies the response code lines (MEM 2.3).

Port Busy (1 line) is sent to the processor when the port buffer cannot accept a request. It is a true signal (MEM 2.8).

ADU Busy (1 line) is sent to the processor when the CMC is writing a 64-bit data word (in response to a read request) into the ADU for later disassembly by the requesting PP. It is a one clock period true signal (MEM 2.6).

SECTION IV-4

DISTRIBUTOR

=====

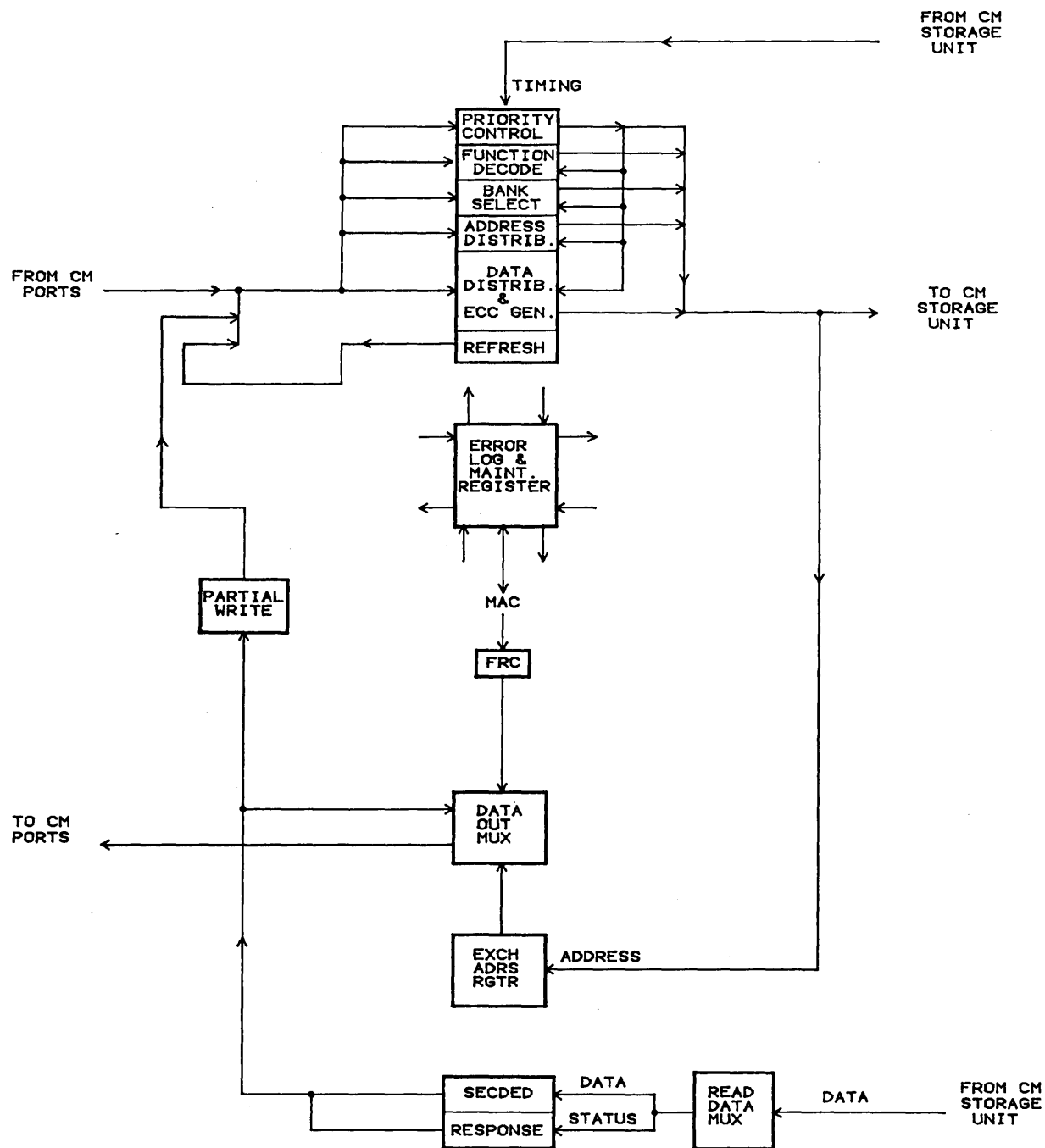
The CM distributor resolves port priority, selects banks, and distributes addresses to an address interface (figure IV-4-1). This interface provides the timing for the memory bank cycles. The distributor also has a refresh control circuit to influence the priority circuit by sending it refresh requests that always have higher priorities than any other requests. It handles maximum data transfer of one word every clock period. This section describes port priority network, memory functions and responses, and its operations.

PORT PRIORITY NETWORK

Port priority is, in order, CPU port then IOU port. If two ports simultaneously request memory banks that are not busy, the requests are honored in the order of priority. There is no long-term lockout of a port by the other port. If a port requests a bank that is busy, the request is not accepted by the priority network (MEM 2.6); instead, a request from a lower priority port for a bank that is not busy is honored. There are two levels of priority, depending on the time of arrival of requests, to prevent long-term lockout of ports. Table IV-4-1 shows the longest period of time that any request waiting in a port buffer register can be locked out.

TABLE IV-4-1. REQUEST LOCKOUT TIME IN BANK CYCLES

Port	Maximum Request Lockout Time in Bank Cycles*	
	Read or Write Requests	Read/Modify/Write Requests
Refresh	1	2
CPU port	2	3
IOU port	3	5
* 1 bank cycle = 8 clock periods = 400 ns		



C0256

Off-sheet references are to CM Ports, figure IV-3-1 and to CM Storage Unit, figure IV-5-1.

Figure IV-4-1. CM Distributor

During a CM request a processor or the IOU transmits write data, an associated memory address, and control bits to one of the CM ports. The port changes the input signal levels from nonstandard to standard levels and then either transfers the data to memory or buffers it. The buffering occurs only in case of a conflict and provides temporary storage. Buffering may result from such conflicts as a request for a bank that is busy or simultaneous requests by two ports. The buffering continues until the conflict clears or the buffer memory is full. When the buffer memory for a port is full, CM does not accept requests from the processor or IOU connected to that port. When the conflict clears, the requests stored in the buffer memory issue one at a time to the distributor.

The distributor may receive one request from each of the two ports at one time. Because the memory can only accept one of the requests at a time, the distributor selects a request on a port priority basis. The priority selection includes a memory refresh request which always takes priority over the ports. The distributor is controlled by the port code bits. These and other control bits originate in the priority translator.

The priority translator (MEM 2.6) resolves conflicts that result from two or more requests for the same memory bank or for simultaneous port requests for the distributor. In addition, the translator evaluates a number of other conditions that can cause conflicts. When none of these is present, the translator provides a signal which allows the ports to continue the transfer of information to the distributor. The translator also determines the availability of the memory banks and starts the bank cycles.

A refresh control circuit (MEM 2.7) influences the priority translator by sending it refresh requests that always have higher priority than any other requests. The control informs the priority translator of a bank refresh with a refresh bank code. The control also determines which addresses within the bank receive the refresh. These addresses transfer directly to the distributor.

The distributor either selects a request from one of the ports or a refresh request, as determined by the priority translator. In either selection, the distributor transfers the address bits to an address interface. This circuit separates the address into component parts to provide individual addresses for the memory arrays. The address interface also provides the timing for the memory bank cycles.

MEMORY FUNCTIONS

Memory performs the following operations as specified by the four-bit code received on the function lines (MEM 2.5). Table IV-4-2 shows the action for a given failure. Refer to Memory Operations for description of each function.

0000	Read
0001	(Not assigned)
0010	Write
0011	(Not assigned)
0100	Read and set lock
0101	Read and clear lock
0110	Exchange
0111	(Not assigned)
1000	CYBER 170 exchange request
1001	Read exchange address
1010	Read free running counter
1011	Refresh counter resync
1100	Interrupt
1101	(Not assigned)
1110	(Not assigned)
1111	(Not assigned)

TABLE IV-4-2. ACTION FOR A GIVEN MEMORY FUNCTION FAILURE

Function \ Error	PW	Data						Mult		
	Path	In	Adrs	Mark	Tag	Fctn	Corr	or	Bound	Illeg
	PE	PE	PE	PE	PE	PE	PE	Read	Fault	Fctn
	PE	PE	PE	PE	PE	PE	PE	PE		
Read		3	3	3	3	2		3		2
Write	1	1	1	1	3	2	4	1	1	2
Read and Set Lock	1	1	1	1	3	2	4	1	1	2
Read and Clr Lock	1	1	1	1	3	2	4	1	1	2
Exchange	1	1	1	1	3	2	4	1	1	2
Read Exch Adrs		3	3	3	3	2				2
Read Free Run Cntr		3	3	3	3	2				2
Refr Cntr Resync						2				2
Interrupt		2			3	2				2

1. Inhibit write.
2. Force single reject response and inhibit writes and interrupts.
3. Operate normally with appropriate response.
4. Correct data read, insert marked bytes, generate new SECDED code, and write modified corrected data back to memory.

MEMORY RESPONSES

The following response codes are sent to a processor in response to function codes (MEM 2.3). Table IV-4-3 shows the response for a given failure.

000	Write response (WR)
001	Write response uncorrectable error (WRUE)
010	Write response corrected error (WRCE)
011	Interrupt response (IR)
100	Read response (RR)
101	Read response uncorrectable error (RRUE)
110	Read response corrected error (RRCE)
111	Reject (REJR)

TABLE IV-4-3. RESPONSE FOR A GIVEN MEMORY FUNCTION FAILURE

Response Function	WR	WRUE	WRCE	IR	RR	RRUE	RRCE	REJR
Read					1	3,4	2	6
Write	1	3,4,5	2					6
Read and Set Lock					1	3,4,5	2	6
Read and Clear Lock					1	3,4,5	2	6
Exchange					1	3,4,5	2	6
Read Exch Address					1	4		6
Read Free Run Counter					1	4		6
Refr Counter Resync				1				6
Interrupt				1				6,7

1. Normal transfer.

2. Corrected error.

3. Multiple or read parity error.

4. Parity error (except function code or tag-in parity error).

5. Bounds fault.

6. Function code parity error or illegal function.

7. Data in parity error.

MEMORY OPERATIONS

REFRESH OPERATION

Central Memory is made up of 64K dynamic memory chips which require the contents to be periodically refreshed. Refreshing a chip is accomplished by supplying just the row address and RAS, so that an entire row of the array in the chip is refreshed at once. A refresh signal from the bank control circuitry to the array paks overrides the chip select to force RAS to all the chips in a bank, so all chips are refreshed at once. The refresh signal is also used to inhibit CAS in the bank control circuitry.

The refresh circuit (MEM 2.7) consists of an interval timer, an address counter, and bank request flip-flops. The interval timer counts at a 1-microsecond rate. Every sixteen microseconds the timer causes all four bank request flip-flops to set. The output of each flip-flop is gated by +Bank Busy, so that if the corresponding bank is available, a refresh request will be made for that bank. The request will be accepted by the distributor immediately, since refresh has highest priority. Only one bank is started at a time, with other banks being started in successive clock cycles (depending on bank availability). In the absence of conflicts, banks are started in the order 1, 3, 0, and 2. Tags, functions and data do not exist for a refresh request; only the address is important, and of the address, only the eight row address bits are used. These bits are supplied by the address counter. The same address is used for all four requests simultaneously. Just after the fourth request is accepted, the address counter increments to point to the next row to be refreshed. For 64K chips there are 256 rows, with two rows being refreshed at once, so it takes 128 times 16 microseconds (about two milliseconds) for a bank to be completely refreshed.

READ OPERATION

When a processor or IOU requests a read (including the selection of the requesting port and the addressing of a memory bank), the bank transfers read data to the data interface. The data transfer includes ECC bits when the memory is operating in the SECDED mode or parity (P) bits when operating in the parity mode. Control circuits within the memory arrays and the data interface provide for the transfer of read/write data on common paths. The data interface also selects the read/write paths from the addressed bank. This selection is necessary because the paths from the nonaddressed banks carry meaningless data back to the data interface.

The SECDED circuit (MEM 2.3, 2.10) generates a new ECC from the write data and compares this ECC with the ECC read from memory. Differences in the comparisons indicate read errors. The SECDED circuit detects both single-bit errors and double-bit errors and corrects the single-bit errors. Parity then replaces the ECC, which is no longer used, and the data becomes corrected read data. This data and parity pass through the partial write select and delay circuit without any modification and then to the distributor.

Port mux select bits (MEM 2.2) control the flow of corrected read data through the distributor. These bits, assigned by the priority translator (MEM 2.6), would arrive ahead of the data except for a delay which occurs in the control delay circuit. The bits direct the distributor to transfer the data to the requesting port. The port changes the read data to nonstandard signal transmission levels before placing it on the lines to the requesting processor or IOU.

The distributor also selects the tag in and function code bits. These bits are present during every read, write, or partial write function performed by the memory. The bits transfer through the ports and distributor in the same way as the address bits. The control delay circuit then changes the tag in bits to tag out bits. The change is in name only. The requesting processor or IOU is the originator of the tag bits which pass through the memory without being used or changed.

In addition to its signal delaying function, the control delay circuit also decodes the function code bits and generates the response code bits. The decoded bits direct the memory to perform one of ten processor- or IOU-initiated functions. The response code bits specify the nature of the response being returned to the processor. The initialization and response of this operation are shown below:

Initialization	Function code, address, and tag are received during one clock period.
Response	Response code, tag, and 64 bits of data as specified by the fullword address.

WRITE OPERATION

If all mark bits are set on a write operation, the selected memory bank performs a write cycle. If any mark bits are cleared on a write operation, the selected memory bank performs a read/modify/write cycle. This operation modifies the bytes in the word specified by the word address and mark bits.

Following a processor or IOU request for a write and the selection of the requesting port, one word of write data transfers from the distributor to the write ECC generator (MEM 2.10). The generator performs a parity check and generates an ECC from the data. If the memory is operating in the SECDED mode, the ECC transfers with the write data into the specified address of the memory bank. If the memory is operating in the parity mode, parity bits transfer with the write data into the memory bank. The initialization and response of this operation are shown below:

Initialization	Function code, address, tag, mark bits, and 64 bits of data are received during one clock period.
Response	Response code and tag are returned.

READ/MODIFY/WRITE (PARTIAL WRITE) OPERATION

This operation modifies the bytes in the word specified by the word address and mark bits. Eight bytes of unmodified data are returned to the processor on these operations.

Following a processor request for a partial write (read/modify/write cycle) and the selection of the requesting port, the request initiates a normal read function. In this case, the read data not only transmits to the processor (as in a normal read function) but it also becomes part of a data modification. This modification takes place in the partial write select and delay circuit and requires write data. The write data transfers from the processor or IOU as in a normal write, except that it changes paths in the data interface. Instead of routing the write data to the memory banks, the data interface routes it to the partial write select and delay circuit (MEM 2.2).

The partial write select and delay circuit delays the arrival of mark bits to coincide with the arrival of both the write and read data. The data modification then takes place. The mark bits (MEM 2.4) define which bytes of the read data receive the modification and other control bits define type of modification. This may be the ANDing, ORing, or exchanging of the read data with the write data on a byte-by-byte basis. The modified data becomes partial write data which transfers to the partial write ECC generator (MEM 2.10).

The partial write ECC generator generates an ECC from the partial write data and transfers it to the data interface with the modified data. The write data and its ECC then write back into the memory bank at the same address originally occupied by the read data. The initialization and response of this operation are shown below:

Initialization	Function code, address, tag, mark bits, and data are received during one clock period.
Response	Response code, tag, and 64 bits of data as specified by the fullword address.

READ AND SET LOCK

This operation forms a logical OR between the marked data-in bytes and the data read from memory, and rewrites the modified data into memory. The data read from memory is returned to the originating element. The initialization and response of this operation are shown below:

Initialization	Function code, address, tag, mark bits, and data are received during one clock period.
Response	Response code, tag, and 64 bits of data as specified by the fullword address.

READ AND CLEAR LOCK

This operation forms a logical AND between the marked data-in bytes and the data read from memory, and rewrites the modified data into memory. The data read from memory is returned to the originating element. The initialization and response of this operation are shown below:

Initialization	Function code, address, tag, mark bits, and data are received during one clock period.
Response	Response code, tag, and 64 bits of data as specified by the fullword address.

EXCHANGE

This operation exchanges the marked data-in bytes with the corresponding bytes in the word read from memory, and rewrites the modified data into memory. The initialization and response of this operation are shown below:

Initialization	Function code, address, tag, mark bits, and data are received during one clock period.
Response	Response code, tag, and 64 bits of data as specified by the fullword address.

CYBER 170 EXCHANGE REQUEST

CM handles CYBER 170 exchange requests from IOU to CPU. The exchange address register is loaded with the address information (MEM 2.1). Data-in bits 59 and 60 give the exchange type; the table below shows how they are used. Data-in bit 63 indicates which CPU is to have the exchange operation (MEM 2.5).

Bit 63 = 0 = first CPU
Bit 63 = 1 = second CPU

<u>Data-In Bits</u>		<u>Function</u>	<u>Exchange Type</u>
59	60		
0	0	Exchange	EXN
0	1	Exchange	MXN
1	0	Exchange	MAN

The initialization and response of this operation are shown below:

Initialization	Function, address, and tag are received during one clock period.
----------------	--

Response A read response and tag are returned. An exchange interrupt and two other signals with the exchange type information are sent to the CPU and remain unchanged until the next CYBER 170 exchange from IOU. The exchange interrupt signal is inhibited on data-in parity error.

READ EXCHANGE ADDRESS

This function is from CPU only for CYBER 170 exchange address. The last PP exchange address is retained in the exchange address register and is accessible by this function. The CYBER 170 exchange signal loads and strobes data into the exchange address register (MEM 2.1). The initialization and response of this operation are shown below:

Initialization Function code and tag are received during one clock period.

Response Response code, tag and 64 bits of data from the 18 bit exchange address register, right justified, zero filled.

READ FREE RUNNING COUNTER

CM reads the output of a counter which guarantees a different value on successive reads. It is common to any element sharing CM and increments every microsecond (MEM 2.1). The initialization and response of this operation are shown below:

Initialization Function code and tag are received during one clock period.

Response Response code, tag, and 64 bits of data from the 48-bit free running counter, right justified, zero filled.

REFRESH COUNTER RESYNC

CM forces a temporary conflict on the requesting port and transmits an interrupt to the port. Clearing of the conflict synchronizes subsequent requests with the memory refresh cycles. This function is a hardware debug tool (MEM 2.7). The initialization and response of this operation are shown below:

Initialization Function code and tag are received during one clock period.

Response Response code and tag. The response code returned is an interrupt response.

INTERRUPT

This function sends an interrupt to the processor specified by the contents of the data received on the data-in lines. Bits are assigned as follows (MEM 2.5):

Bit 61 Sends interrupt to the first CPU.

Bit 63 Sends interrupt to the second CPU.

The initialization and response of this operation are shown below:

Initialization Function and data are received during one clock period.

Response A single interrupt response is returned (MEM 2.5). One or more ports may receive an interrupt due to an interrupt operation. This includes sending an interrupt to itself.

SECTION IV-5

STORAGE UNIT

=====

Access to the storage unit has a data path 64 bits wide and is accompanied by eight bits of error correcting code/parity. Address and control signals are accompanied by parity bits.

The storage unit consists of memory arrays with data, address, and control logic interfaces (figure IV-5-1).

The storage unit with 64K-chip arrays is expandable from 2 to 4, 8, 12, and 16 megabytes. 256K-chip arrays allow expansion to 16 or 32 megabytes. The memory has two banks for 2 megabytes and four banks for all other memory sizes. It is a dynamic random access memory. Maximum data transfer rate is one word every clock period for the CPU port and one word every two clock periods for the IOU port. A read or write bank cycle time is eight clock periods or 400 ns and a read/modify/write bank cycle time is 16 clock periods or 800 ns.

MEMORY ARRAYS AND ADDRESSING SCHEME

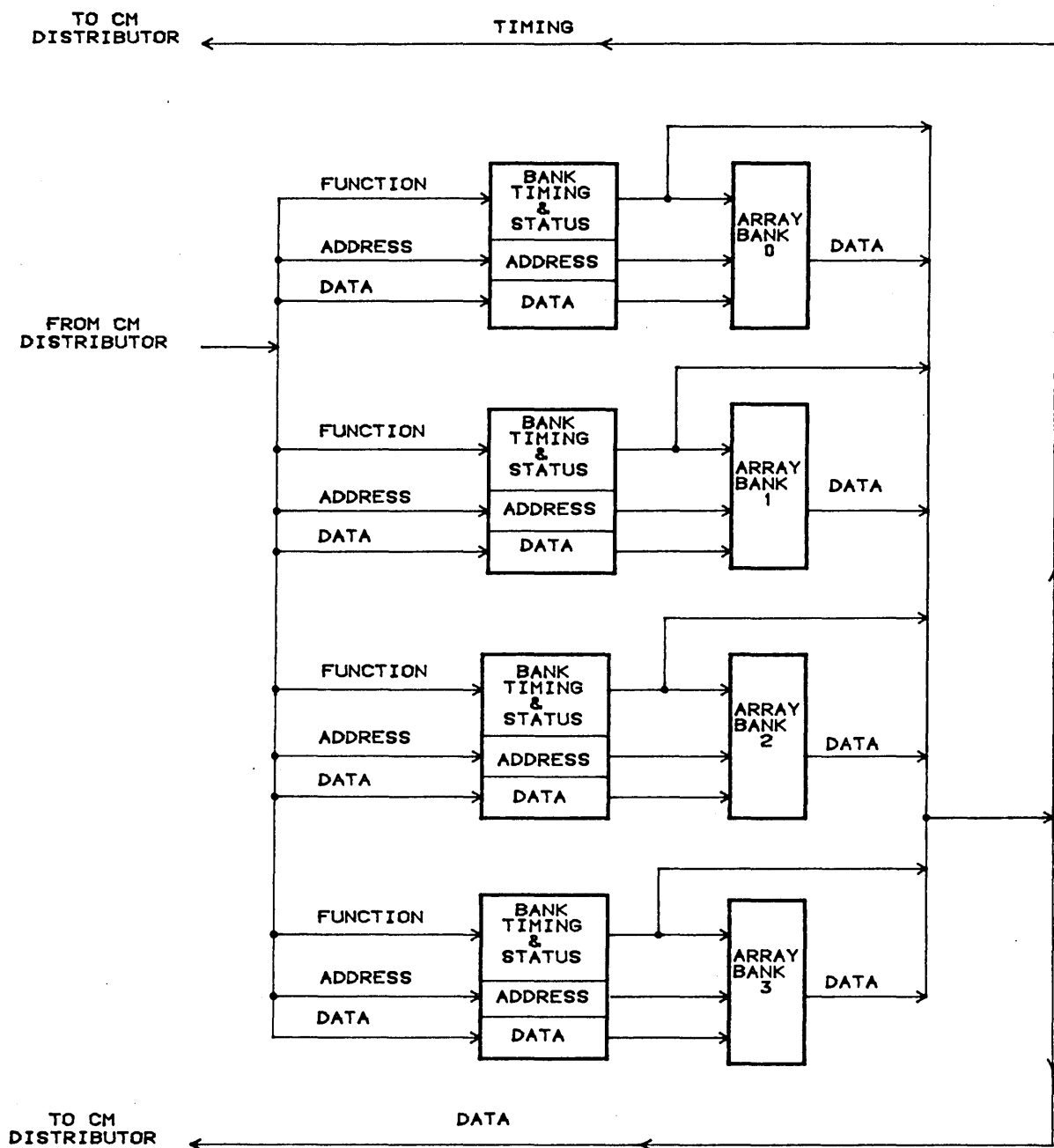
CM operates in interleaved (phased) mode of addressing. In interleaved mode, sequential addressing results in phased banks (e.g. address 0000 bank 0, 0001 bank 1, 0002 bank 2, 0003 bank 3, 0004 bank 0, 0005 bank 1, 0006 bank 2, 0007 bank 3, and so on).

The 64K memory chips are arranged in two or four banks with one to four boards per bank and two chip groups per board (figure IV-5-2). In a two-MB configuration, address bits 44 to 59 define the row and column address of the array with bits 43 and 60 being the chip select and bank select bits respectively. In other memory configurations, address bits 43 to 58 define the row and column address of the array with bit 42 being the chip select bit and bits 59 and 60 being the bank select bits. Bits 40 and 41 are the pak select bits.

ADDRESS WRAP-AROUND

If address exceeds physical memory and is less than maximum memory size, write is at nonexistent address and read is all ones (assuming no error report; maximum memory is sixteen MB). If the address exceeds maximum memory size, read/write are at the contents of wrap address.

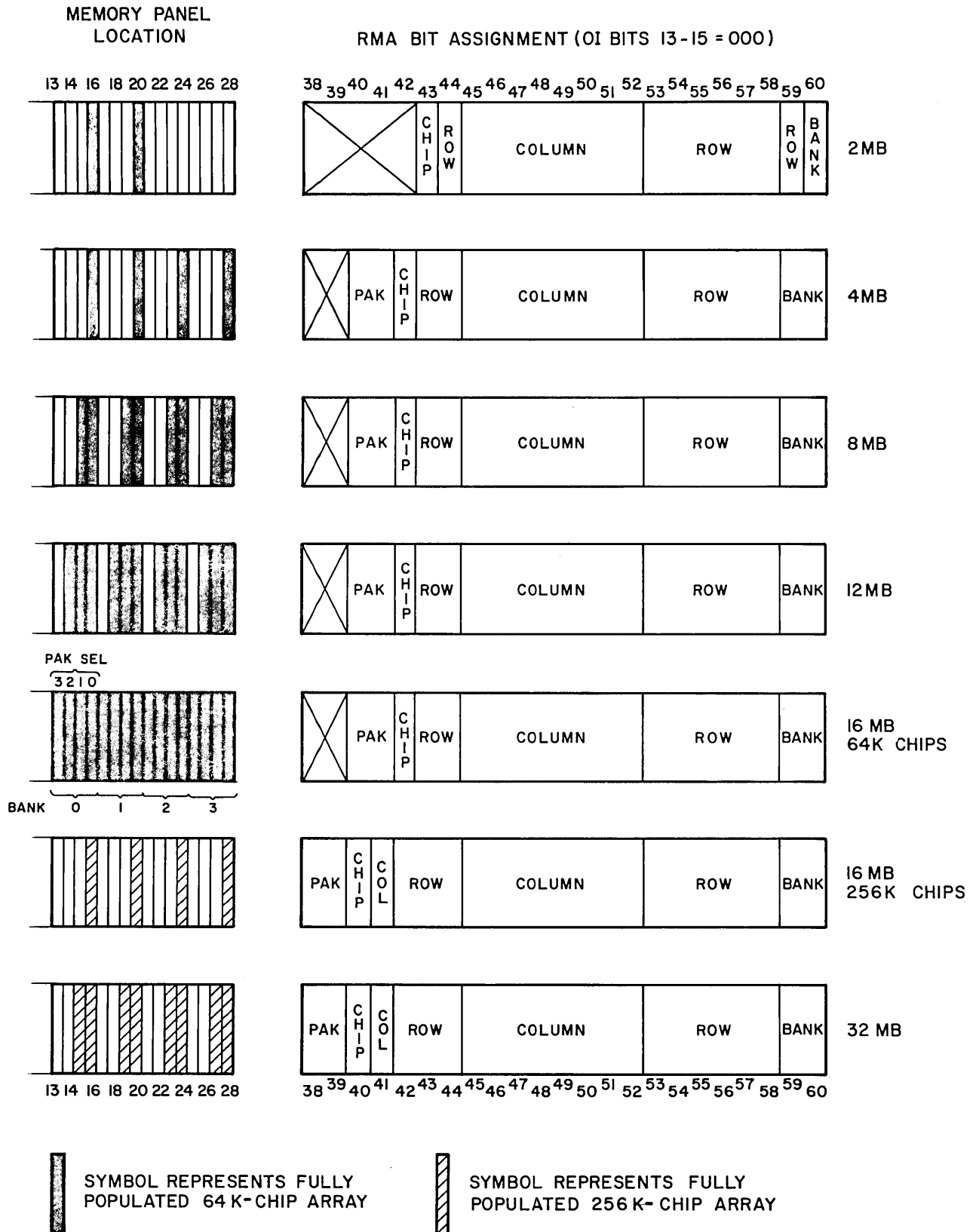
If address referenced is in portion of memory configured out of system, read/write are at the contents of wrap address.



C0255

Off-sheet references are to CM Distributor, figure IV-4-1.

Figure IV-5-1. CM Storage Unit



CI065

Figure IV-5-2. Memory Array Addressing Scheme (Interleaved Mode)

CM RELIABILITY, ACCESSIBILITY, AND MAINTAINABILITY FEATURES

PARITY

All address, control, and data paths that constitute the port interfaces carry a parity bit for each eight-bit byte except ADU data which carries a parity bit for each four-bit half byte. All major address, control, and data paths that are internal to the memory and do not carry SECDED code carry a parity bit for each eight-bit byte or four-bit half byte.

MAINTENANCE REGISTERS

Table IV-5-1 lists the memory maintenance registers and their access privilege. Maintenance registers fall into two classifications: those accessible via the maintenance channel, and those accessible via memory ports. Refer to the Maintenance and Parts Manual for their descriptions.

The free running counter is a 48-bit incrementer accessible by memory ports. It increments at a one microsecond rate. Successive reads of the free running counter guarantee different values.

TABLE IV-5-1. MEMORY REGISTER ACCESS PRIVILEGES

Register Name	Memory Port Access Privilege		Maintenance Chan. Access
	Read	Write	
Status Summary	No Access	No Access	Read
Element ID	No Access	No Access	Read
Options Installed	No Access	No Access	Read
Environment Control	No Access	No Access	Read/Write
Bounds Register	No Access	No Access	Read/Write
Correctable Error Log	No Access	No Access	Read/Write
Uncorrectable Error Log 1	No Access	No Access	Read/Write
Uncorrectable Error Log 2	No Access	No Access	Read/Write
Free Running Counter	Unprivileged	No Access	Write

SECDED

All data within the memory banks is protected with SECDED logic. During a write, eight error correction code (ECC) bits or check bits are generated prior to storing data in memory. The word stored is 72 bits long (64 data plus eight ECC bits). During a read, the data plus check bits pass through the SECDED logic where single and multiple bit errors are detected and single bit errors are corrected.

The SECDED logic detects an error by regenerating new check bits from the 64 bit word read from memory and comparing the new check bits with the check bits read from memory. Eight syndrome bits are generated from this comparison of check bits and represent an error condition if the check bits do not compare.

The read data bytes that came into the DD pak fan out from the first read data register to a syndrome generator (MEM 2.1). The generator produces partial parity and parity signals to create a new ECC in the parity checker. This checker compares the new ECC with the old ECC which comes with the read data. The new-to-old ECC comparison creates syndrome code bits S0 through S7.

When none of the code bits are present, a no error condition is indicated. When an even number of code bits are present (MEM 2.3), a double-bit error (DBE) is indicated. When an odd number of code bits are present, a single-bit error (SBE) is indicated. Table IV-5-2 shows the SECDED parity check matrix. Table IV-5-3 gives the syndrome codes and the corrected data bits.

One set of syndrome code bits enters the corrected error log register (MR 2.12). This is one of nine registers accessible by the maintenance channel.

A second set of syndrome code bits enters the error correction section of DD (MEM 2.1). This section decodes the syndrome code bits and determines, in the case of a single-bit error, which bit and byte are in error. The bit and byte corrections AND together to form correction bits which enter the exclusive OR gate in the read data section. The correction bits toggle the OR gate of the bit in error, correcting a single-bit error in the incoming read data.

The occurrence of a single-bit error also requires correction of the parity bit for the byte that contains the corrected bit. This correction occurs in the same way as for the read data; the incorrect parity bit is toggled.

BOUNDS REGISTER

This register has three bits reserved for the bit vector for port bounds (MR 2.15), 16 bits reserved for upper bounds, and 16 bits reserved for lower bounds as shown below (MR 2-18):

0	2 3	31 32	47 48	63
Bit Vector For Port Bounds	Not Used	Upper Bounds	Lower Bounds	

Bits 0, 1, and 2 designate CPU 0, IOU, and CPU 1 ports respectively. Bits 3 through 31 are not used and are always read as zeroes. The upper two bits of upper (bits 32 and 33) and lower (bits 48 and 49) bounds are also not used. Setting a bit in the bit vector for port bounds confines the corresponding port: writes are inhibited for all addresses greater than or equal to the upper bounds or less than the lower bounds; read operations are not affected.

TABLE IV-5-2. CM SECDED PARITY CHECK MATRIX

		BYTE 0								BYTE 1								BYTE 2								BYTE 3										
SB	CB	0	1	2	3	4	5	6	7		8	9	10	11	12	13	14	15		16	17	18	19	20	21	22	23		24	25	26	27	28	29	30	31
UPPER SYNDROME BIT	S0	64				•				64	•		•		•		•	64				•				64	•			•		•		•		•
	S1	65		•						65			•				•	65		•						65			•					•		
	S2	66				•				66								66				•				66				•						
	S3	67					•			67	•	•				•		67					•			67	•			•				•		
LOWER SYNDROME BIT	S4	68	•	•		•		•		68			•					68	•	•		•		•		68				•						
	S5	69			•				•	69		•						69			•				•	69		•								
	S6	70								70			•					70				•				70				•						
	S7	71		•				•		71				•				71	•			•				•	71					•				

		BYTE 4								BYTE 5								BYTE 6								BYTE 7										
SB	CB	32	33	34	35	36	37	38	39		40	41	42	43	44	45	46	47		48	49	50	51	52	53	54	55		56	57	58	59	60	61	62	63
UPPER SYNDROME BIT	S0	64		•						64	•	•		•		•	•	64		•						64	•	•		•		•		•		•
	S1	65				•				65			•				•	65				•				65			•					•		
	S2	66						•		66								66						•		66				•						
	S3	67				•				67	•	•				•		67				•				67	•			•				•		
LOWER SYNDROME BIT	S4	68	•	•		•		•		68	•	•	•	•				68	•	•		•		•		68	•	•	•	•						
	S5	69			•				•	69			•					69			•				•	69				•						
	S6	70								70				•				70				•				70					•					
	S7	71		•				•		71			•					71	•			•				•	71					•				

- NOTES:
- CHECK BITS 64 THROUGH 71 ARE ODD PARITY FOR BITS MARKED ☒ IN THE CORRESPONDING HORIZONTAL LINES.
 - SYNDROME BITS S0 THROUGH S7 ARE PARITY ERRORS FOR THE CORRESPONDING CHECK BITS.

C1066

TABLE IV-5-3. SYNDROME CODES/CORRECTED DATA BITS

CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT	CODE	BIT
0 0	NONE	2 0	(1) 66	4 0	(1) 65	6 0	(2)	8 0	(1) 64	A 0	(2)	C 0	(2)	E 0	X 0
0 1	(1) 71	2 1	(2)	4 1	(2)	6 1	(3)	8 1	(2)	A 1	(3)	C 1	(3)	E 1	(4) 0
0 2	(1) 70	2 2	(2)	4 2	(2)	6 2	(3)	8 2	(2)	A 2	(3)	C 2	(3)	E 2	(4) 16
0 3	(2)	2 3	(3)	4 3	(3)	6 3	(2)	8 3	(3)	A 3	(2)	C 3	(2)	E 3	X 16
0 4	(1) 69	2 4	(2)	4 4	(2)	6 4	(3)	8 4	(2)	A 4	(3)	C 4	(3)	E 4	(4) 2
0 5	(2)	2 5	(3)	4 5	(3)	6 5	(2)	8 5	(3)	A 5	(2)	C 5	(2)	E 5	X 2
0 6	(2)	2 6	(3)	4 6	(3)	6 6	(2)	8 6	(3)	A 6	(2)	C 6	(2)	E 6	X 18
0 7	X 44	2 7	(4) 60	4 7	(4) 46	6 7	X 62	8 7	(4) 45	A 7	X 61	C 7	X 47	E 7	(4) 18/63
0 8	(1) 68	2 8	(2)	4 8	(2)	6 8	(3)	8 8	(2)	A 8	(3)	C 8	(3)	E 8	(4) 1
0 9	(2)	2 9	(3)	4 9	(3)	6 9	(2)	8 9	(3)	A 9	(2)	C 9	(2)	E 9	X 1
0 A	(2)	2 A	(3)	4 A	(3)	6 A	(2)	8 A	(3)	A A	(2)	C A	(2)	E A	X 17
0 B	X 12	2 B	(4) 28	4 B	(4) 14	6 B	X 30	8 B	(4) 13	A B	X 29	C B	X 15	E B	(4) 17/31
0 C	(2)	2 C	(3)	4 C	(3)	6 C	(2)	8 C	(3)	A C	(2)	C C	(2)	E C	X 3
0 D	X 40	2 D	(4) 56	4 D	(4) 42	6 D	X 58	8 D	(4) 41	A D	X 57	C D	X 43	E D	(4) 3/59
0 E	X 8	2 E	(4) 24	4 E	(4) 10	6 E	X 26	8 E	(4) 9	A E	X 25	C E	X 11	E E	(4) 19/27
0 F	(2)	2 F	(3)	4 F	(3)	6 F	(2)	8 F	(3)	A F	(2)	C F	(2)	E F	X 19
1 0	(1) 67	3 0	(2)	5 0	(2)	7 0	X 36	9 0	(2)	B 0	X 4	D 0	X 32	F 0	(2)
1 1	(2)	3 1	(3)	5 1	(3)	7 1	(4) 36	9 1	(3)	B 1	(4) 4	D 1	(4) 32	F 1	(3)
1 2	(2)	3 2	(3)	5 2	(3)	7 2	(4) 52	9 2	(3)	B 2	(4) 20	D 2	(4) 48	F 2	(3)
1 3	(3)	3 3	(2)	5 3	(2)	7 3	X 52	9 3	(2)	B 3	X 20	D 3	X 48	F 3	(2)
1 4	(2)	3 4	(3)	5 4	(3)	7 4	(4) 38	9 4	(3)	B 4	(4) 6	D 4	(4) 34	F 4	(3)
1 5	(3)	3 5	(2)	5 5	(2)	7 5	X 38	9 5	(2)	B 5	X 6	D 5	X 34	F 5	(2)
1 6	(3)	3 6	(2)	5 6	(2)	7 6	X 54	9 6	(2)	B 6	X 22	D 6	X 50	F 6	(2)
1 7	(4) 44	3 7	X 60	5 7	X 46	7 7	(4) 54/62	9 7	X 45	B 7	(4) 22/61	D 7	(4) 47/50	F 7	X 63
1 8	(2)	3 8	(3)	5 8	(3)	7 8	(4) 37	9 8	(3)	B 8	(4) 5	D 8	(4) 33	F 8	(3)
1 9	(3)	3 9	(2)	5 9	(2)	7 9	X 37	9 9	(2)	B 9	X 5	D 9	X 33	F 9	(2)
1 A	(3)	3 A	(2)	5 A	(2)	7 A	X 53	9 A	(2)	B A	X 21	D A	X 49	F A	(2)
1 B	(4) 12	3 B	X 28	5 B	X 14	7 B	(4) 30/53	9 B	X 13	B B	(4) 21/29	D B	(4) 15/49	F B	X 31
1 C	(3)	3 C	(2)	5 C	(2)	7 C	X 39	9 C	(2)	B C	X 7	D C	X 35	F C	(2)
1 D	(4) 40	3 D	X 56	5 D	X 42	7 D	(4) 39/58	9 D	X 41	B D	(4) 7/57	D D	(4) 35/43	F D	X 59
1 E	(4) 8	3 E	X 24	5 E	X 10	7 E	(4) 26/55	9 E	X 9	B E	(4) 23/25	D E	(4) 11/51	F E	X 27
1 F	(3)	3 F	(2)	5 F	(2)	7 F	X 55	9 F	(2)	B F	X 23	D F	X 51	F F	(2)

X SINGLE BIT ERROR
CORRECTED① SYNDROME CODE BIT
FAILED (SINGLE CODE
BIT SET)② DOUBLE ERROR OR
MULTIPLE ERROR(EVEN
NUMBER OF CODE
BITS SET)③ MULTIPLE ERROR REPORTED
AS SINGLE ERROR④ DOUBLE ERROR OR
MULTIPLE ERROR WITH
INDICATED BIT(S)
INVERTED

The lower 14 bits of the upper bounds (bits 34 through 47) or lower bounds (bits 50 through 63) represent bits 38 through 51 of the real memory address (RMA). The contents of the bounds register do not affect access to the maintenance registers.

MEMORY CONFIGURATION SWITCH REGISTER

This is a one-byte register which is only accessible to the deadstart microprocessor (MR 2.18). It allows logical reconfiguration of memory to remove failing memory portions from the address space and its contents may be changed by entering commands from the console under the deadstart microprocessor maintenance display.

Configuration switch register bits 2 through 7 represent the positions of configuration switches* SW3, SW4, and SW5 as shown in table IV-5-4.

TABLE IV-5-4. MEMORY CONFIGURATION SWITCH REGISTER

64K-chips	Bit	Description	Action
	0,1	(Not Used)	
	2	SW 3 Up	RMA bit 40 forced to 1
	3	SW 3 Down	RMA bit 40 forced to 0
	4	SW 4 Up	RMA bit 41 forced to 1
	5	SW 4 Down	RMA bit 41 forced to 0
	6	SW 5 Up	RMA bit 42 forced to 1
	7	SW 5 Down	RMA bit 42 forced to 0

When 256-K chips are used, the Memory Configuration Switch Register will be interpreted as follows:

256K-chips	Bit	Description	Action
	0,1	(Not Used)	
	2	SW 1	(Not used)
	3	SW 1	(Not used)
	4	SW 2 Up	RMA bit 39 forced to 1
	5	SW 2 Down	RMA bit 39 forced to 0
	6	SW 3 Up	RMA bit 40 forced to 1
	7	SW 3 Down	RMA bit 40 forced to 0

* These are not physical switches.

TABLE IV-5-5. MEMORY CONFIGURATION

Installed Memory	Failing Portion of Memory	Memory Configuration Switches	Options Installed Register Bits 16-22	Remaining Available Memory
	RMA Bit 38 39 40 41 42	SW 1 2 3 4 5	12 16 17 18 19 20 21	
4 MB (64K chip)	X X X X 0 X X X X 1	- - - - U - - - - D	0 1 0 0 0 0 1 0 0 0 0 0 0 1	2 MB 2 MB
8 MB (64K chip)	X X X 0 X X X X 1 X	- - - U - - - - D -	0 1 0 0 0 1 0 0 0 0 0 0 1 0	4 MB 4 MB
16 MB (64K chip)	X X 0 X X X X 1 X X	- - U - - - - D - -	0 1 0 0 1 0 0 0 0 0 0 1 0 0	8 MB 8 MB
16 MB (256K chip)	X X 0 X X X X 1 X X	- - U - - - - D - -	1 1 0 0 1 0 0 1 0 0 0 1 0 0	8 MB 8 MB
32 MB (256K chip)	X 0 X X X X 1 X X X	- U - - - - D - - -	1 1 0 1 0 0 0 1 0 0 1 0 0 0	16 MB 16 MB

If both bits for a particular switch are set, the effect is to force a one.

Table IV-5-5 shows reconfiguration for a memory using 64K chip (maximum memory size is 16 MB).

Further reconfiguration of a 16 MB system may be performed by setting SW4. For example, a 16 MB system that has been reconfigured to eight MB by SW3 may be further degraded to four MB by using SW4. OI register bits 19, 20, and 21 determine the remaining available memory as follows:

<u>Bit</u>	<u>Remaining Memory</u>
18	16 MB
19	8 MB
20	4 MB
21	2 MB

After reconfiguration, addresses greater than the remaining available memory either wrap-around into available memory or reference nonexistent memory as specified by the configuration switch register.

SECTION V-1

INTRODUCTION

=====

The central processing unit (CPU) and its interfaces with central memory (CM) and maintenance access control/maintenance channel are shown in figure V-1-1. The CPU contains the following functional units which perform address translation, arithmetic, logic, and control operations:

- Control store loads, dumps, sweeps, and executes microprograms. Random access memory holds the microprogram that controls internal CPU operations, reads and decodes micrands, and generates control signals for the functional units in CPU. Control store also provides micrand addressing facilities. The maintenance channel (MCH - channel 178) drives the maintenance access control (MAC) which provides a data path to load or store the control store.
- Execution units under microcode control perform logical, arithmetic, shifting, and addressing functions using operands supplied by register file or CM. Random access register files hold process state registers, constant values, and operands for processing. They also provide temporary storage space for intermediate results during instruction execution.
- Read next instruction (RNI) unit contains instruction buffers and control logic to prefetch sequential instruction words from CM prior to execution. It selects sequential instructions from these words according to the decoded instruction length and issues them to the control store for execution. The unit increments the program address register (P register) according to the decoded instruction length. The execution units also use the RNI unit for stream operations.
- MAP translates virtual addresses into real memory addresses and also performs address validity checking. It performs access validation during translation. The MAP receives PVAs from execution hardware or the RNI hardware for translation. Translation uses tables in CM which are accessed via the CM interface. Translation occurs by microcode routines leaving the translation results in the MAP buffers. If simultaneous translation requests are made, addresses from the execution hardware have priority.

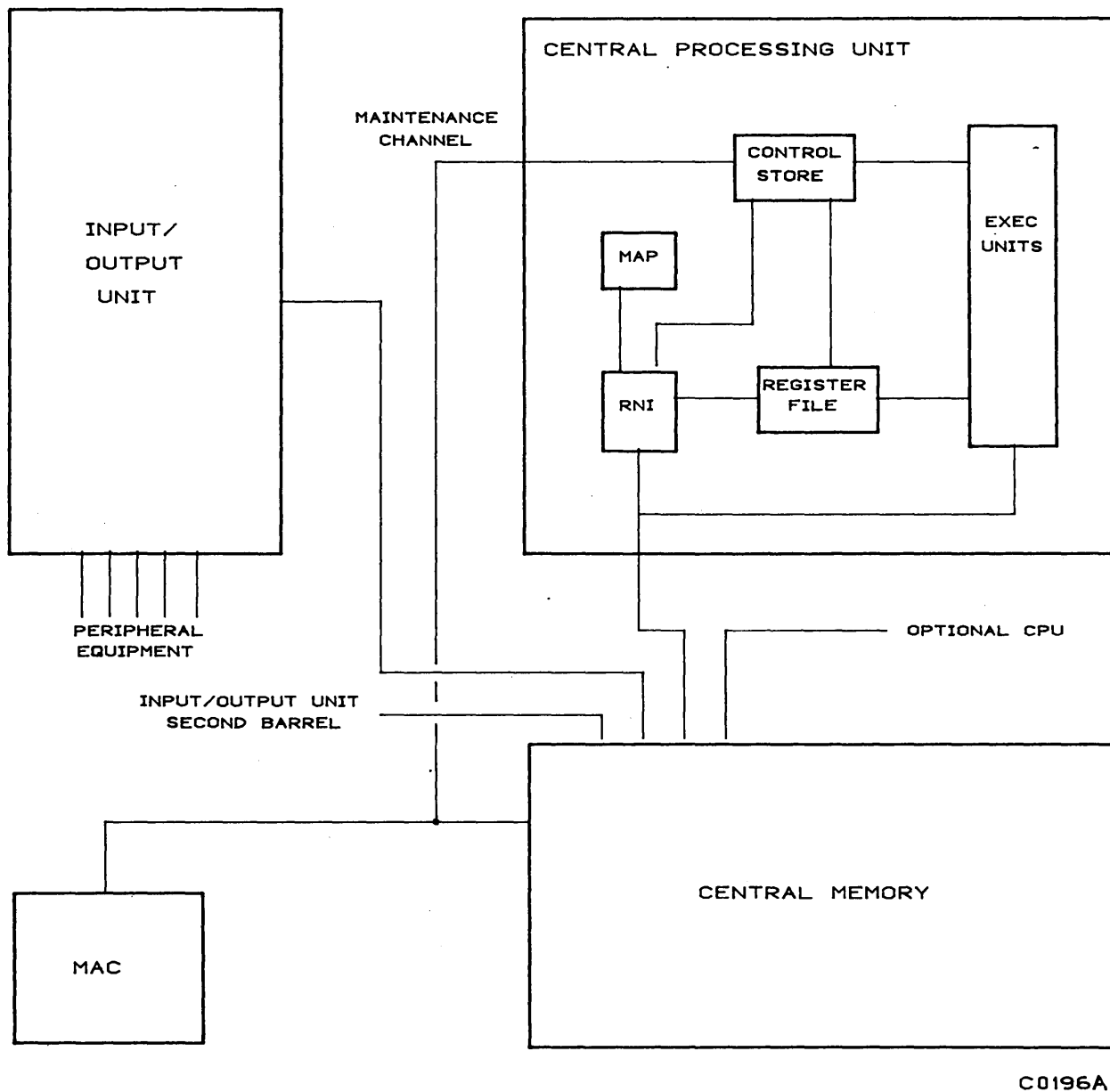


Figure V-1-1. Computer System Models 810 and 830 Block Diagram

SECTION V-2

CONTROL STORE

Control store stores the CPU controlware (also known as microcode or micrands) and the micrand address selection logic (MAS) as shown in figure V-2-1. This writable control store contains an interface with the maintenance access control (MAC) through which the micrands are loaded from a permanent storage. Control Store includes some of the model dependent processor registers.

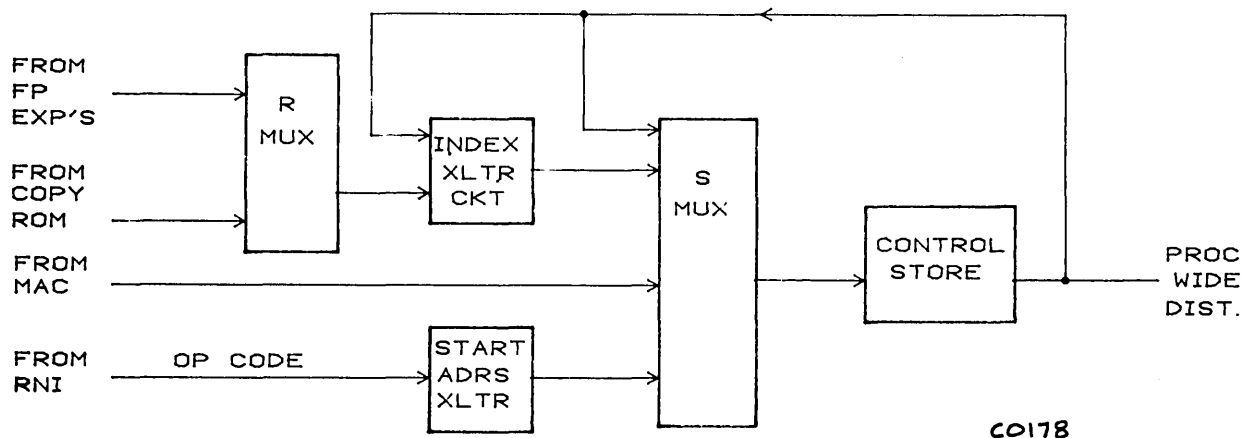


Figure V-2-1. Control Store

HARDWARE DESCRIPTION

The hardware consists of the following:

- Writable control store (CS)
- Micrand holding register (MHR)
- Control store address register (S register)
- Control store address multiplexer (S mux)
- Return register 1 (R1 register)
- Return register 2 (R2 register)
- Micrand address selection logic (MAS)
- Page offset adder (R adder)
- R multiplexer (R mux)
- Model dependent CPU registers:
 - Processor fault status (PFS)
 - Retry corrected error log (Retry CEL)
 - MAP corrected error log (MAP CEL)
 - Processor test mode (PTM)
 - Processor identifier (PID)
 - Breakpoint register (BKPT register)

WRITABLE CONTROL STORE

Control Store Word

The structure of a control store word is shown in figure V-2-2.

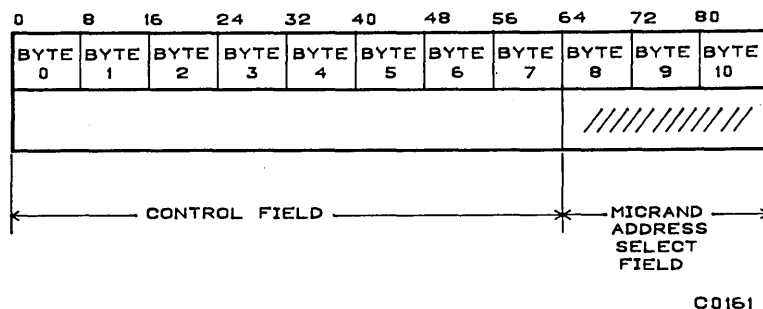


Figure V-2-2. Control Store Word

Each control store word has 82 data bits and 11 parity bits (one odd parity bit per byte). The 82 data bits are divided into a 64-bit control field and an 18-bit micrand address select field. The control field is subdivided into fields which control the execution of instructions. (See section V-3, Execution Units, and appendix E.)

The micrand address select field controls control store addressing (explained later in this section). Bits 69, 70, and 84 through 87 are used. No memory exists for the unused bits which are always logic zero.

Control Store Organization

The control store contains a single 8K word memory in five paks. Four paks are pak type 1DR0, in logic chassis locations CP016, CP017, CP018, and CP019, each containing 16 data bits plus 2 parity bits. The 1DS0 pak type, in logic chassis location CP020, contains 18 data bits plus 3 parity bits.

NOTE

No physical memory exists for the unused bits in control store (that is, bits 69, 70, and 84 through 127).

Control Store Timing

The control store operates on the standard 20-MHz clock of the CPU. Its memory chips have a maximum access time of 42 ns. The micrand read cycle time of 100 ns includes time to generate and to select the micrand address based on information in the micrand address select field and signals from the CPU.

execution unit. At the end of the micrand cycle the data is available in the micrand holding register (MHR).

The control store is loaded through the MAC by the maintenance access hardware (MAH). The timing for writing control store depends on the MAH.

MICRAND HOLDING REGISTER (MHR)

The MHR (CS 2.2 and PIP 2.0) holds the current micrand until the next micrand is available. The MHR contains 77 data bits (bits 0 through 63 and 71 through 83) and 10 parity bits.

CONTROL STORE ADDRESS REGISTER (S REGISTER)

The S register (CS 2.2) is a 13-bit register. It holds the current micrand address until the CPU requires the next micrand. This register functions as an address incrementer if the next address is the current address plus one. The micrand address selection logic controls the load or increment function of this register except when the control store is being loaded by MCU through MAC.

CONTROL STORE ADDRESS MULTIPLEXER (S MUX)

The S mux (CS 2.1) allows the selection of the next micrand address (other than S + 1). It selects one of the following:

- Code 0 is BRA (bits 0 through 4) concatenated with R adder (bits 5 through 12)
- 1 is BRA (bits 0 through 12)
- 2 is R1 (bits 0 through 12)
- 3 is R2 (bits 0 through 12)
- 4 is S (bits 0 through 4) concatenated with BRA (5 through 12)
- 5 is micrand address for next instruction
- 6 is MAC data
- 7 is trap entry address

Code 0 selects the branch address from micrand bits 71 through 75 concatenated with the output of the R adder. With the R mux and R adder functions, this select code handles MAS of A/B/C/D/E/F.

Code 1 selects the branch address from micrand bits 71 through 83. When MAS is 1 or 4, this code is used.

Code 2 selects R1. When MAS is 2, this code is used.

Code 3 selects R2. When MAS is 3, this code with branch condition logic is used.

Code 4 selects S register bits 0 through 4 concatenated with branch address from micrand bits 76 through 83. When MAS is 6 or 7, this code is used.

Code 5 selects 12 bits of micrand address for next instruction from CPU. When MAS is 5, this code is used.

Code 6 selects 13 bits of data from MAC for use as the deadstart address or the starting address for write/read control store or data pattern to test the S register.

Code 7 selects the trap entry address.

RETURN REGISTER 1 (R1 REG), RETURN REGISTER 2 (R2 REG)

R1 and R2 (CS 2.1 and CS 2.2) each hold 13 bits of address for the micrand return addresses. When bit 68 of the current micrand is set and S + 1 is in the R copy of the S register, S + 1 is loaded into R1. Therefore, micrand bit 68 is remembered for one additional clock time so that R1 can be strobed one clock time after the S register is strobed. When enabled by the MAS code, R2 is strobed at the same time as the S register.

MICRAND ADDRESS SELECTION LOGIC (MAS)

MAS (CS 2.2) decodes micrand bits 64 through 68. It receives control signals from the RNI unit, branch conditions from the execution unit, CYBER 170 exchange signals, and trap control signals. It generates the necessary select for the R mux, S mux, and branch condition mux. It not only controls the R adder functions and the S register functions but also enables the strobing of the R1, R2, and S registers.

PAGE OFFSET ADDER (R ADDER)

This 8-bit adder (CS 2.1) calculates the page offset of a micrand address in a 64-word page. It receives two operands--one from the R mux and the other from micrand bits 76 through 83. The result is sent to the S mux to form the 13-bit micrand address.

R MULTIPLEXER (R MUX)

The R multiplexer (CS 2.1) allows the selection of the following:

- Copy read-only memory (ROM)
- Register file A data (AD) bits 56 through 63 left-shifted by one
- AD bits 56 through 63
- Floating point ROM

The micrand address selection logic controls the R mux to select one of the above for MAS equals A/B/C/D/E/F. It also controls R mux to select AD bits 56 through 63 for the R adder operation of BRA bits 5 through 12 plus AD bits 56 through 63.

MODEL DEPENDENT CPU REGISTERS

The functions and bit assignments of these registers are described elsewhere in this manual. MAC can access these registers directly. The codes which select each register are shown in table V-2-1.

TABLE V-2-1. MODEL DEPENDENT CPU REGISTERS

Function	CS Data Bytes											S Register		BKPT Register	
	0	1	2	3	4	5	6	7	8	9	10	6	7	6	7
SEL UN	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1
Byte code 4	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1
Byte code 2	0	0	1	1	0	0	1	1	0	0	1	0	0	1	1
Byte code 1	0	1	0	1	0	1	0	1	0	1	0	0	1	0	1
SEL UM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SEL UM register 2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SEL UM register 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

	Maintenance Scan				PFS Register				MAP CEL	PID Register	PTM Register
	4	5	6	7	4	5	6	7	7	7	7
SEL UN	0	0	0	0	0	0	0	0	0	0	0
Byte code 4	1	1	1	1	1	1	1	1	1	1	1
Byte code 2	0	0	1	1	0	0	1	1	0	0	1
Byte code 1	0	1	0	1	0	1	0	1	0	1	0
SEL UM	1	1	1	1	1	1	1	1	1	1	1
SEL UM register 2	1	1	1	1	0	0	0	0	0	0	0
SEL UM register 1	0	0	0	0	1	1	1	1	0	0	0

HARDWARE OPERATIONS

CONTROL STORE ADDRESSING

The rightmost 20 bits of each micrand are in the form shown in table V-2-2 and figure V-2-3.

TABLE V-2-2. CONTROL STORE ADDRESSING

Bit(s)	Field	Description
64-67	MAS	Controls the source of the next micrand.
68	R1	Controls whether or not the return register R1 is strobed with S address + 1.
69-70		Dummy unused bits.
71-83	BRA	13-bit BRA field allows branching anywhere in control store.
71-75	BCOND	5-bit field defines one of 32 possible branch conditions.
76-83	PAGE OFFSET	8-bit field describes an address within a 256 location segment of the control store. The current segment in the leftmost 5 bits of the S address register defines a particular segment.

64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87
MAS		R1	/////	BRANCH ADDRESS																//////////			
MAS		R1	/////	BRANCH COND.				PAGE OFFSET								//////////							

C0162

Figure V-2-3. Micrand Address Select

MAS FieldMAS CodeNext Micrand Address

- | | |
|---|--|
| 0 | S + 1 is the next sequential address. |
| 1 | Branch unconditionally to address defined by BRA field. |
| 2 | Branch to address held in return register R1 if the branch condition defined by the BCOND field is true. |
| 3 | Branch to address held in return register R2 if the branch condition defined by the BCOND field is true. |
| 4 | Branch unconditionally to 13-bit BRA address and instruct the execution unit that the leftmost 64 bits of the micrand are emitted data to be held in the micrand diagnostic holding register (MDHR), also known as the EMIT 64 register. |

MAS CodeNext Micrand Address

- | | |
|---|---|
| 5 | Instruction exit if the branch condition defined by BCOND field is false. Instruction exit allows the next micrand address to come from the next instruction's opcode, a pretrap, or a MAC entry point. |
| 6 | Branch conditionally to a two-part address: the leftmost five bits from the current S value and the rightmost eight bits from bits 76 through 83 of the page offset. The branch occurs if the defined branch condition is true. |
| 7 | As in MAS equals 6 except that branch occurs if defined condition is false. |
| 8 | Go to next sequential micrand. Strobe the return register R2 with the leftmost five bits of the BRA field and the sum of the eight rightmost bits of the BRA field and the eight rightmost bits of the execution units AD register. |
| 9 | Go to next sequential micrand, S +1, and strobe R2 with bits 76 through 83 of the BRA field. |
| A | Branch to an address composed of the leftmost five bits of BRA field and the rightmost eight bits of BRA plus the rightmost eight bits of the AD register. |
| B | As in MAS equals A except the AD register's rightmost eight bits shift left one place zero inject before the 8-bit add occurs. |
| C | As in MAS equals A except the rightmost eight bits of the BRA field are added to a 5-bit quantity from the exponent arithmetic section called the FP ROM. |
| D | As in MAS equals C except eight bits of output from the COPY ROM are added. |
| E | As in MAS equals A except the AD rightmost eight value is subtracted from the BRA field's rightmost eight. |
| F | As in MAS equals B except the function is subtract. |

NOTE

If MAS code is 0 and R1 is 1, CPU halts. If MAS code is 5 and R1 is 1, CPU does instruction exit. However, the next address forms from the 13-bit BRA field of the previous micrand with the least significant bit of the BRA field ORed with the most significant bit of the next instruction's opcode. This operation is called a flaky exit. It accesses business data processing (BDP) descriptors where the MSB is the descriptor F bit.

BCOND Field

The BCOND field specifies which hardware condition in CPU decides if a branch is taken. The 32 branch conditions are shown in table V-2-3.

TABLE V-2-3. BRANCH CONDITION

BCOND	Mnemonic	Condition Selected
0	MET	Branch condition is forced true.
1	IBS	Integer/Boolean unit condition is used.
2	RES	Result bit 0 FF.
3	FL1	FLAG number 1.
4	FL2	FLAG number 2.
5	BCM	BDP sense condition.
6	NOT	Branch condition is forced false.
7	LJ=	LENJ counter equals zero.
8	LC=	LENC counter equals zero.
9	LK=	LENK counter equals zero.
A	GLO	Use MAP condition.
B	IPL	CPU is in the 64-bit state mode.
C	MON	CPU is in monitor mode.
D	TES	CPU is in test mode.
E	LJ-	LENJ counter decrements after equalling zero.
F	JSG	BDP J sign FF.
10	SD=	BDP D mux output equals zero.
11	AEX	A side FP unpacker detects a defined zero exponent.
12	BEX	B side FP unpacker detects a defined zero exponent.
13	QGE	LENJ counter equals zero or the BDP Q counter is greater than or equal to 19; used in decimal-to-binary conversion.
14	STA	Stack purge flag is set.
15	OBI	Badder bit zero is a one.
16	FL3	Flag 3 is set.
17	PTS	Page table search flag is set.
18	TRA	Trap flag is set.
19	SPF	Scan in progress flag is set in the debug mask register.
1A	ELF	End of list flag is set.
1B	MAC	(Not Used)
1C	FPE	A floating point exception.
1D	SAD	Sadder sense condition.
1E	DEB	Debug is active
1F	NPM	PMF request

LOADING CONTROL STORE

MCU loads microcode into control store through MAC. The following explains control store hardware functions.

The LDS0 pak, logic chassis location CP020, receives the starting address where data is loaded via the MAC data bus. Two MAC data bus cycles load the starting address into S register. S register bits 0 through 7 are loaded first, then bits 8 through 12.

For each control store word, control store receives eleven writes from MAC data bus, writing one byte of data plus one parity bit each time. The receiving pak checks parity. When parity error is detected, the error flag and the associated PFS bits set. Parity error does not inhibit a write operation.

After writing one word, control store receives a MAC Incr S signal which increments S register by one. This repeats until MCU finishes loading control store. In each word, the MCU sends 16 bytes. MAC does not write bytes 11 through 15 into control store. The MAC sends zeros to the MCU for bytes 11 through 15 when MCU reads control store.

START/STOP CONTROL STORE

MCU sends functions to MAC to start/stop control store. The start control store signal sets the Run flag in control store and, if S Full has been set, S1 Full sets and a micrand loads into MHR. The micrand is available to CPU and control store is ready for the next micrand. At the end of the current instruction, the halt control store signal resets the Run flag which stops control store from sending the next micrand to execution unit.

NORMAL SEQUENCE

Deadstart

The MCU sends a function to MAC to deadstart CPU. Control store functions are:

- To load S register with an entry point
- To start a CS cycle similar to the functions described earlier in this section

MAS logic prepares the next micrand address. The CS is ready to start the next micrand cycle. It continues to execute new cycles when the Run flag is set and the execution unit is ready for the next micrand. At the end of processor initialization, the microcode puts the control store in instruction exit state ready to accept the instruction's opcode, a pretrap, or a MAC entry point.

Instruction Exit

When bits 64 through 67 of the current micrand have a code of 5, the MAS generates S mux select code equalling 5. If the branch condition is false, the instruction's opcode is loaded into S register.

When control store is in instruction exit state, it allows microdetour and MAC entry points. When in 60-bit mode, the instruction is on word boundary and can allow the 60-bit mode state exchange entry points. When these conditions occur simultaneously, the priorities are as follows:

1. Microtraps
2. 16-bit mode exchange microdetour
3. MAC request control store microdetour

Breakpoint

When EC bit 36 is set, the control store stops at the micrand address equal to the breakpoint address in the BKPT register plus one.

Sweep Mode

By setting EC bits 24, 36, and 37 and sending a start function from MCU, the control store increments the micrand address until breakpoint hit occurs. This checks for parity errors in the control store and for proper working order in the breakpoint circuit and the micrand address incrementer.

Microstep

By setting EC bit 5 and sending a start function from MCU, the control store starts at the current micrand address and stops at the next micrand address.

NOTE

Exercise care when using microstep. Signals such as central memory response may be lost when stepping to the next micrand.

Instruction Step

By setting EC bit 31 and sending a start function from MCU, the control store starts at the current micrand address and stops at the next instruction exit micrand.

MAINTENANCE FUNCTIONS

PARITY

An odd parity bit exists for each byte of data stored in control store and each micrand address byte sent from LDS0 pak to the four LDRO paks. Parity is checked every micrand cycle. An error detected in a pak sets the error flag and the associated processor fault status (PFS) bit. If EC bits 6 and 24 are set, bit 61 of the PFS summary is set.

When MCU writes or reads control store or any of the model dependent processor registers, parity is checked for each byte transferred through the MAC data bus. An error detected during the MAC data bus cycle causes the MAC to transmit an error signal and a ready signal to the MCU.

READ CONTROL STORE VIA MCH

Content of control store reads back through the MAC and MCU for verification. This ensures good micrands are stored in control store and provides an alternate path to test each memory cell in control store.

MAINTENANCE SCAN

Maintenance scan can read a set of signals from CPU after a number of clock intervals (table V-2-4). A write function from MCH with type code 4, register number equals 00, loads the binary number of clock intervals into the maintenance scan limit counter. A read function from MCH with type code 4, register number equals 00, reads the set of signals from CPU.

TABLE V-2-4. MAINTENANCE SCAN

Byte	Bit	Signals	Origin
0	0	CS Halt	8UNO
	1	S Full	8UNO
	2	S1 Full	
	3	M Full	
	4	M1 Full	
	5	CYBER Exchange Request	
	6	Not Next Instruction	
	7	Select Trap	

CONTROL STORE ERROR LOG

CSEL is a 64-bit register. It is accessible using address 81 or 91. Bytes 3-7 store error information for each of the five pak locations as follows:

<u>Byte Number</u>	<u>Pak Type</u>	<u>Pak Location</u>	<u>Pak Description</u>
3	DR	CP-16	DR1
4	DR	CP-17	DR2
5	DR	CP-18	DR3
6	DR	CP-19	DR4
7	DS	CP-20	DS

The five bytes of the error log operate independently of each other. When the error log is read, any byte that is non-zero indicates that an error has occurred on that pak. If the most significant bit (DBE) of a particular byte is set, the error is uncorrectable. If any of the other bits are set, a corrected error will occur. All bytes share the same format as follows:

<u>Bit Number</u>	<u>Description</u>
0	Double Bit Error Flag (DBE): 0 = no DBE 1 = uncorrected error has occurred
1	Chip Select: 0 = lower chip failed 1 = upper chip failed
2-7	Syndrome Code: Code identifies which correctable bit failed.

For a corrected error, the SYNDROME CODE and CHIP SELECT point to a specific failing-chip. MCR bit 62 and SS bit 62 will be set. This information will remain until a CLEAR ERRORS is issued.

For uncorrected errors, the DBE bit of the CSEL register as well as PFS bit 63, MCR bit 48 and SS bit 61 will be set. This condition will cause an interrupt to the CPU.

The operating system will record the condition of registers 80 and 81 if SS bit 61 is set. It will record registers 90, 91 and 93 if SS bit 62 is set. It will then issue CLEAR ERRORS and restart CPU. Lastly, it will check SS bits for clear state.

There is no provision for writing the error log via MAC. For diagnostic purposes, the error log can be sufficiently tested by forcing single and double errors through patterns loaded into Control Store.

SECTION V-3

EXECUTION UNITS

=====

The execution units provide hardware to perform arithmetic/logical operation and business data processing under micrand control as defined in appendix E. This section outlines the functions performed by the execution units, the register files, and the primitive formation unit.

The following execution units and the units which interface with them are described in this section:

- 64-Bit Unit (BAD 1.0, 2.0-2.2, 3.0)
- 18-Bit Unit (SAD 1.0, 2.0, 3.0)
- Byte Unit (BDP) (BDP 1.0, 2.0-2.4, 3.0)
- Streaming Unit (STR 1.0, 2.0-2.2, 3.0)
- Register Files Unit (RF 1.0, 2.0-2.7, 3.0-3.1)
- Trap Control Unit (CS 1.0, 2.0-2.3, 3.0-3.1)
- Primitive Formation Unit (PIP 1.0, 2.0-2.2, 3.0)

Micrands control the execution of instructions. Figure V-3-1 shows the relationship between the S, micrand holding register (MHR), primitive, and write time of a micrand. During the S time of a micrand, its address is held in the S register. At MHR time, the micrand that was addressed is held in the MHR register. For a complete list of instruction execution times see appendix F.

All the execution units use the micrand bits at MHR time to input their respective primitive registers. The register file read and the shift count select (SCS) field are the only fields active during MHR time of a micrand. At the beginning of every micrand's primitive time, the AD and BD registers are ready for input. The execution occurs during primitive time. The result is selected by the write data multiplexer (WD mux) and is sent either to the MAP as an address, to the CSU as data, or to the WD register for storage in the register file.

EXECUTION UNITS DESCRIPTIONS

The CPU contains four execution units: 64-bit unit, 18-bit unit, byte unit, and stream unit. AD and BD hold the input data of each unit during the primitive time of the unit being functioned. The micrand format controls which execution unit is active.

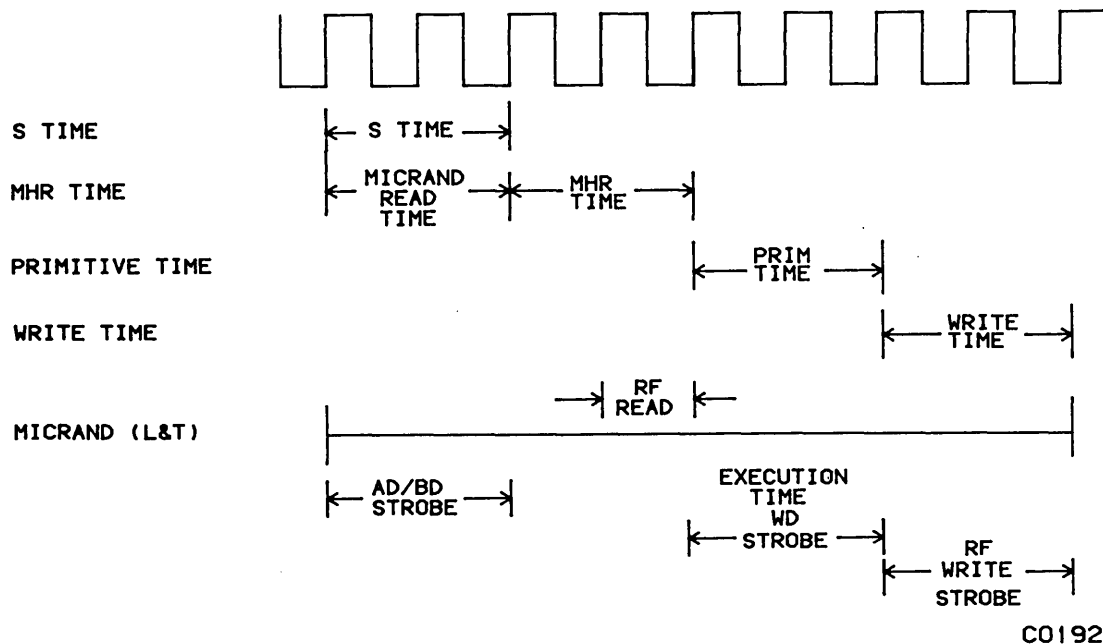


Figure V-3-1. Relationship between S, MHR, Primitive, and Write Times of a Micrand

64-BIT UNIT

Also known as the B adder or big adder, the 64-bit unit performs arithmetic operations (figure V-3-2). Shown on BAD 1.0, 2.0, and 3.0, it is organized on four 1DF0 boards and is partitioned exactly the same as the RF (two bytes per board). Associated micrand fields are described in appendix E. The 64-bit unit contains the ASX mux, SBD mux, B adder, shifter, and execution sense field.

ASX Mux (Register File A Data Sign Extended Mux)

Fed by AD register, the ASX mux is a 64-bit mux (BAD 2.0) which forms the A data input for the B adder. ASX mux enables sign extension.

SBD Mux (Register File B Data Mux)

Fed by the BD register and forming the B data input for the B adder, SBD mux is a 64-bit mux (BAD 2.0) which provides paths for multiply/divide and shift operations.

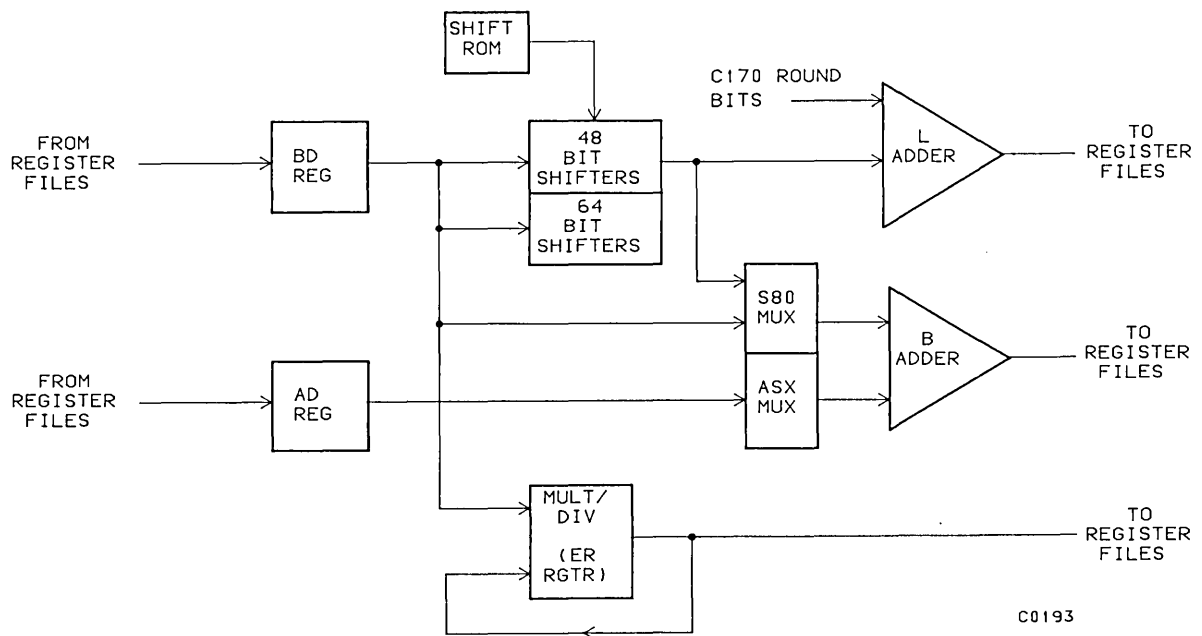


Figure V-3-2. 64-Bit Adder Circuit

B adder

B adder is a 64-bit adder (BAD 2.0) which performs 64-bit arithmetic in ones and twos complement modes. During format B, the execution sense conditions are derived from the B adder.

Shifter

The shifter (BAD 2.2) forms part of the 64-bit data path and is fed by the BD register only. During right end-off shifts, up to 48 bits of the shifted-off data become the data at the B input of the L adder (BAD 2.2) which is used in the CYBER 170 mode for FP and rounding operations. The 64 most significant shifter output bits are selectable (sel code 0) at the SBD mux. Although the output of the shifter is 96 bits wide, the input actually is only 64 bits wide. See appendix E for shifter functions.

Execution Sense Field (ESC)

The execution sense field, micrand bits 42 to 45 in formats A, E, and B, selects conditions under which the control store can branch. These conditions occur during the primitive time of the micrand and effect the address of S register at S time of the micrand.

18-BIT UNIT

The 18-bit unit includes the S adder (SAD 2.0), which is used for 18-bit (60-bit mode) address arithmetic, 12-bit (60-bit mode) exponent arithmetic, 16-bit (64-bit mode) exponent arithmetic and miscellaneous operations such as calculation of a shift count shown in figure V-3-3.

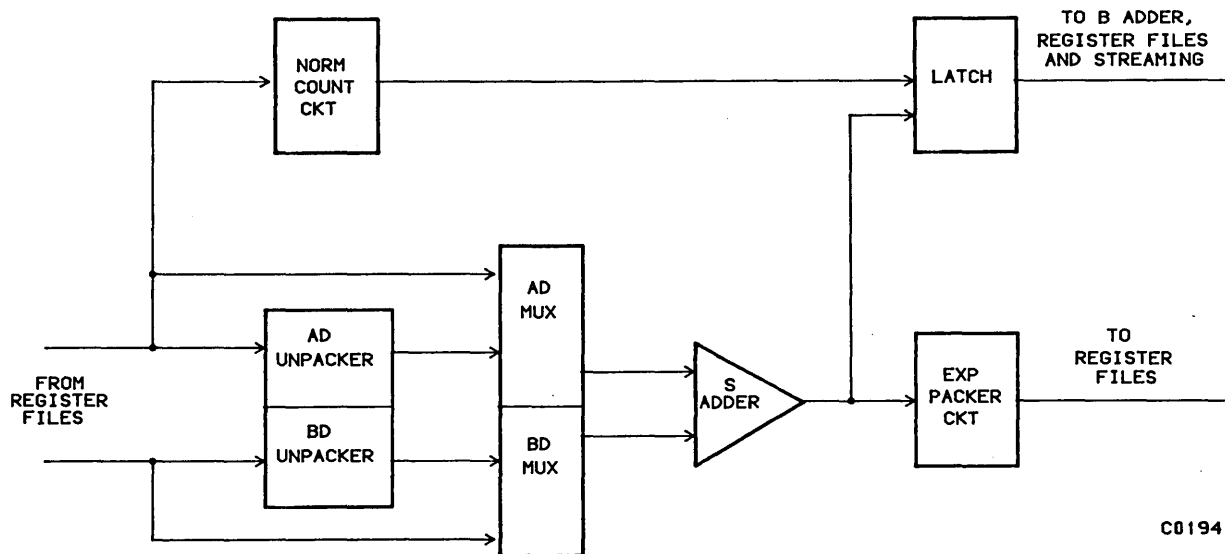


Figure V-3-3. 18-Bit Adder Circuit

Exponent Unpackers

The exponent unpackers (figure V-3-4), depending on the mode of the processor, produce unpacked exponent(s) suitable for exponent arithmetic within the S adder. During 60-bit mode, the unpacker uses six bits of exponent sign extension (two bits during 64-bit mode).

If the unpackers are not selected, the rightmost 18 bits of AD or BD are copied unaltered to S adder inputs. Either unpacker output can be forced to all ones under certain conditions. Refer to Unpacker Control, appendix E.

18-Bit Adder

The 18-bit S adder operates in ones or twos complement logic. The adder function is under microcode control during format A or E operation (see appendix E). During format A or E, the execution sense conditions derive from the S adder (SAD 2.0) on 8TC0, logic chassis location CP010.

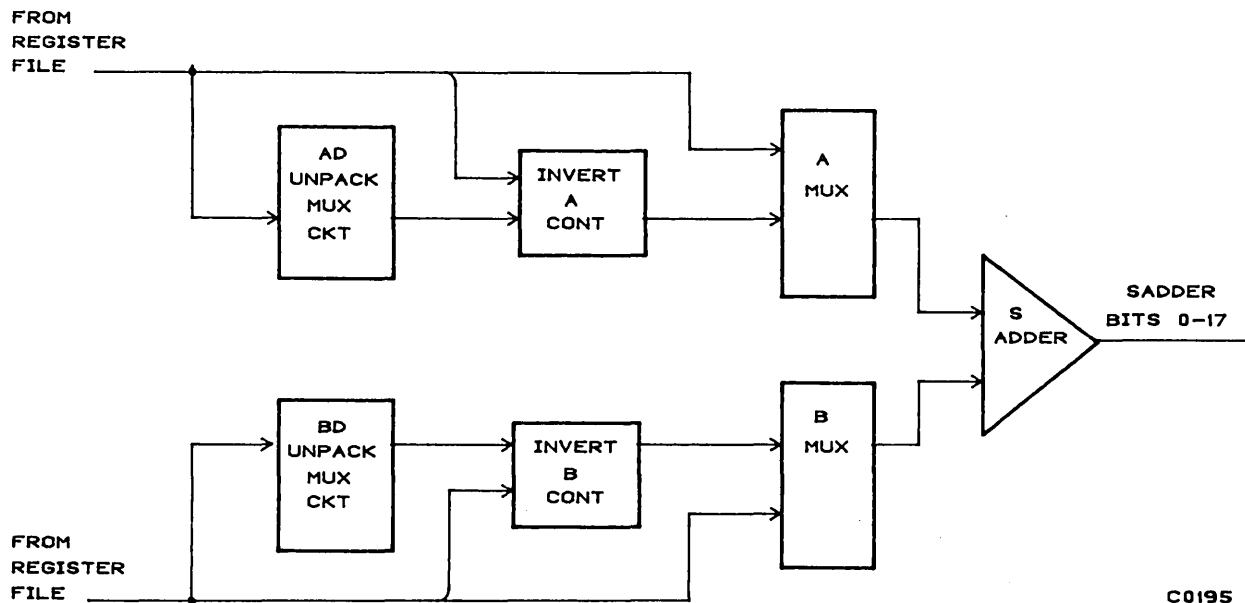


Figure V-3-4. Unpacker Circuits

Normalizer

The normalizer generates a shift count by examining the 64-bit value in AD and by making it available for saving in the S adder latch. The saved result can be used to shift the operand into a normalized position in a later micrand. The normalization count is that count which would, when applied as a left shift to a 64-bit integer, cause bit 0 or bit 16 (in a 48-bit co-efficient) to change. Normalization is used during floating point operations.

18-Bit Latch

The 18-bit latch (SAD 2.0) selectively latches either S adder output or six bits of normalization count (right-justified). The latch provides an input to the shift count select (SCS) mux and the write register through ARVI mux and WD mux.

Shift Count Select Mux

The 7-bit SCS mux uses one of eight sources for its shift count register. SCS bits 49 through 51 control the SCS (see appendix E) and are valid only during format C or B. During other formats, the shift count is forced to zero. The shift count select is active at MHR time only.

Exponent Packer

The 16-bit exponent packer network (SAD 2.0) forms a packed exponent according to the FP formats for 60- and 64-bit modes (figure V-3-5). S adder supplies the raw unpacked exponent.

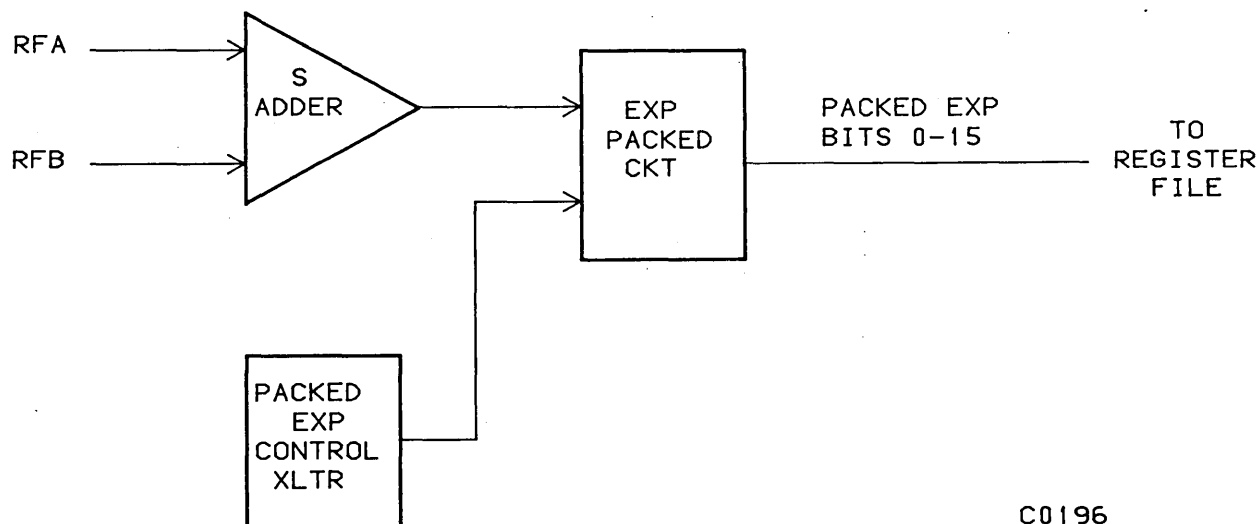


Figure V-3-5. Exponent Packer Circuit

During 64-bit mode packs, bit 3 of S adder is inverted to form the bias bit position of the exponent (exponent bit 1). The coefficient sign is bit 0 of the exponent. S adder bits 4 through 17 inclusive are copied directly to exponent bit positions 2 through 15. The packed 16-bit (64-bit mode) exponent is passed to the ARVI mux (RF 2.0).

For 60-bit mode packs, S adder bits 8 through 17 are copied to exponent bit positions 6 through 15. Adder bit 7 is inverted to form a CYBER 170 bias bit position (exponent bit 5). The coefficient sign is extended from bit 0 through bit 4. The 12-bit (60-bit mode) exponent is right-justified within the 16-bit sign-extended exponent when passed to the ARVI mux. If the coefficient sign of the result is negative (execution sense condition A=1), the entire 16 bits are ones complemented by the packer. The final packed exponent is then concatenated with the coefficient in the ARVI mux to produce the FP result.

BYTE UNIT

The byte unit moves, compares, adds, and subtracts decimal data fields. It also moves and compares binary data fields to execute business data processing instructions. The byte unit consists of the AD/BD disassembly, writing the register file from XBD, data ROM and mux, byte scan mux, decimal operations, and compare operations.

AD/BD Disassembly

The RJB and RKB muxes (BDP 2.0) select one byte of AD and BD respectively. The J3 and K3 counters control the RJB and RKB mux selects. Refer to Process Multiple Bytes in this section.

RGA=01000 or 01001 controls the direction in which J3 counts. RGB=01000 or 01001 controls the direction in which K3 counts. For initialization of these counters, see J3, K3, and C3 Control in appendix E.

Writing the Register File from XBD

The XBD mux (BDP 2.3) decides the function the unit executes and selects the data written into RF. The byte from XBD is broadcast across the WD register through ARVI 1, ARVI 2, and the WD mux (RF 2.0). The C3 counter controls the RF partial write and selects the byte from XBD into the correct RF byte. (See Processing Multiple Bytes later in this section). RGC=01000 or 01001 controls the direction in which C3 counts. For initialization of the C3 counter, refer to J3, K3, and C3 Control in appendix E.

Data ROM and Mux (Preprocess Path)

The data ROM and data mux (BDP 2.1) preprocess source data fields. RJB addresses the data ROM through RJL. The data ROM provides sign information: BDP digit and sign validity, the combined Hollerith digit, and the translated (EBCDIC - ASCII) byte. The data mux selects the data from the data ROM or RJL.

Checking for all digits equals zero is performed on the output of the data mux when the XBD field = 0101, 0110, or 1001. The J sign FF or the K sign FF may be forced positive according to the JFX or KFX fields if all the digits are equal to zero.

Byte Scan Mux

The byte scan mux uses the three LSBs of RJL to select one bit of RKL. The scan hit signal is enabled with XBD=1110. A scan hit causes RJL to latch.

Decimal Operations

Add/Subtract

During an add or subtract micrand (XBD=1100), RJC selects either the four LSBs of RJL or their nines complement depending on the PAS2 field (see appendix E). RKC selects the four LSBs of RKL. A 4-bit decimal add is performed on the data in RJC and RKC with the carry from the addition stored and added to the next decimal digit. A zero or nine check, depending on whether RJC selected RJL or the nines complement, checks for overflow from the addition.

Binary to Decimal Convert

During a binary to decimal convert, micrand (XBD=1111), the binary source byte, is in RJL. The data mux selects either the four MSBs of RJL or the four LSBs of RJL for conversion from binary to a decimal digit with a carry (4-bit conversion greater than nine). The four LSBs of RKB are multiplied by sixteen in the X16 network. The carry from the 4-bit binary to decimal conversion is added into the X16 network one digit later. The X16 result, greater than 100 circuit, produces a carry which is added to the X16 network two digits later. The X16 network produces two decimal digits, D1 and D2. During the first iteration of the convert, RJC selects the result from the 4-bit binary to decimal conversion. During the following iterations, RJC selects the delayed D1 and RKC selects D2. The contents of RJC and RKC are added together in the decimal adder. Overflow checking is not performed on the result.

XAO Mux, CHC ROM, and E ROM (Post-Process Path)

The XAO mux assembles a packed data type destination field, injects the ASCII zone character, and provides a binary data path through XAO to the overflow detectors. Refer to the heading XAO Mux in appendix E.

If XBD=0110 and the E mode FF are set, the XBD mux selects the E ROM. The E ROM translates the byte from XAO from ASCII to EBCDIC.

The six bits of CHC ROM address come from the four LSBs of XAO, the result sign, and a decode of XBD=1011. When XBD does not equal 1011, the CHC ROM provides the combined Hollerith character. When XBD=1011, the CHC ROM provides the unpacked separate sign hexadecimal 2B (positive), hexadecimal 2D (negative).

Compare Operations

Decimal Compare

During a compare micrand (XBD=0010), providing the XBDP field = 001, RJC selects either the four LSBs of RJL or their nines complement depending on the PAS2 field. RKC selects the four LSBs of RKL. The compare result FFs are set according to the result of the addition.

Byte Compare

The data in RJL and RKL are compared for binary magnitude. The results of the compare are strobed into the compare result FFs. A byte miscompare causes RJL and RKL to latch.

STREAM UNIT

The stream unit improves performance in areas of the machine where repetitive operations occur. This unit controls such operations while they occur. The main streaming operations are as follows:

- Load multiple registers from CM.
- Store multiple registers in CM.
- Process multiple bytes for the BDP.

Load Multiple Registers from CM

Under the control of RGC (RF 2.5), the address of the first word loaded is stored in the J or K address counter. Each time the CSU is ready with more data, the J or K address counter increments. The number of words loaded is kept in the YKWA counter. When YKWA reaches zero, the operation ends. Only full words can be loaded into RF.

If RGC says JK.48 during the loading of the A registers, no data is written into the top 16 bits of RF. The J counter is used for the address of the A registers and the K counter is used for the address of the X registers.

Store Multiple Registers in CM

Under the control of RGA, the address of the first word to be stored is loaded in J or K address counter. Each time CSU is ready to accept more data, the J or K counter increments. The number of words stored is kept in YKWA counter. When YKWA reaches zero, operation terminates. When RGA is equal to 6X, the store of the first and last words is accompanied by the appropriate partial write information.

When RGA is equal to 4X while storing the A registers, zeros fill in the upper 16 bits of the ASX mux. During the storage of multiple registers, the J counter is used for the RF address of the A registers and the K counter is used for the RF address of the X registers.

Process Multiple Bytes for the BDP

RGA, RGB and RGC are set to either 8X or 9X during BDP operations that involve multiple moves of data through the BDP. See section titled Micrand Fields RGA, RGB, RGC. On the signal from the BDP, J3, K3, or C3 advance in the specified direction. When the byte of a word is read or written, the appropriate counter updates. J3 and K3 keep track of the byte address which disassembles the bytes and C3 keeps track of which partial to set. The J, K, and C length counters keep track of the respective byte lengths during the operation and terminate the operation on reaching zero.

When the appropriate control bit is a one during the primitive time of format F (STR 2.0), J3 or K3 or C3 is strobed with the data now latched into BNR3, the starting byte address of a data field.

The YKWA counter keeps track of either the number of words accepted from CM on a load multiple transfer or the number of words sent on a store multiple transfer.

The YKWR counter keeps track of the number of requests the RNI, containing the CM data field address, makes to the MAP. When YKWR reaches zero, the execution unit sends a signal to the RNI that stops the request.

Load store multiple length (LSM) is a calculated length derived from the stack frame descriptor and is equal to the number of A's plus the number of X registers loaded or stored. Refer to appendix E.

LDB is a length calculated from the load or store length determination box. The calculation network determines the number of words transferred when the leftmost address and the number of bytes are known. This network also calculates the first and last word-mark lines for storing multiple registers in CM. Refer to appendix E.

EXECUTION UNIT INTERFACE DESCRIPTIONS

REGISTER FILE UNIT

The RF unit (RF 1.0, 2.0-2.7, 3.0-3.1) is organized on four 1DF0 logic boards shown in figure V-3-6. The main and directly related hardware components of the RF unit are as follows:

RFA	256- by 72-bit RAM which forms an input to the AD mux.
RFB	256- by 72-bit RAM which forms an input to the BD mux.
AD mux	72-bit multiplexer under the control of the RGA and ADS micrand fields (appendix E). The AD mux is the input to the AD register.
BD mux	72-bit multiplexer under the control of RGB micrand field and the BDS select (appendix E). The BD mux is the input to the BD register.
AD register	72-bit register which holds the A data for the execution units.
BD register	72-bit register which holds the B data for the execution units.
WD multiplexer	72-bit multiplexer selecting data from all the execution units. The WD mux is the input to WD register, CM, and MAP.
WD register	72-bit register, fed by WD mux, holds the write data for RF.

Reading and Writing RF

Each of the two register files, RFA and RFB, has a 256- by 72-bit RAM. Each of four boards (1DF0, logic chassis locations CP003, CP004, CP005, and CP006) has 2 bytes of data and an odd parity bit associated with each byte.

RFA and RFB files are readable during the MHR time of a micrand and writable during the primitive plus time of a micrand. The read address may be specified independently but, during the write time, both RFA and RFB select the same write address.

RF write data is held by the write data (WD) register which is strobed at the end of the primitive time of each micrand. At the end of a micrand's MHR time, the AD and BD registers are strobed with the RF data selected by the AD and BD muxes. This data is available to the execution units during the entire primitive time of the micrand.

Micrand Fields RGA, RGB, RGC

The RGA, RGB and RGC micrand fields control the reading and writing of RF. Refer to appendix E for a detailed explanation.

RGA is a 9-bit micrand field located in micrand bits 4 through 12 in all formats which controls, at MHR time, the address read from RFA.

RGB is a 9-bit micrand field located in micrand bits 13 through 21 in all formats which controls, at MHR time, the address read from RFB.

RGC is a 9-bit micrand field located in micrand bits 22 through 30 in all formats which controls, at MHR time, the address written into the WDA register at primitive time. Refer to the heading Reading and Writing RF in this section.

MICROTRAP MECHANISM

The conditions which initiate the microtrap mechanism occur individually or in combination. All of them temporarily inhibit further execution of micrands.

When microtrap condition occurs, the taking a trap FF sets, the primitive good FF clears, the catchable enable FF clears, and the non-catchable enable FF clears. Along with these actions, the miscellaneous conditions are disabled and the retry in progress FF sets when the retry enable FF is set.

Once a microtrap begins, all the necessary flip-flops of the machine are reinitialized and the micrand pipe is restarted at the micrand address 16X, where X is the address specified by the micrand trap address ROM. When the trap occurs, the block of the register file write enables is active and the P register is blocked from incrementing.

When the micrand from the control store is ready for execution, the taking a trap FF clears and all the flip-flops that retained the trap condition are cleared. The processor then resumes its normal operation.

Catchable Conditions

Conditions which set the following MCR and UCR bits are called catchable conditions:

<u>Catchable Condition</u>	<u>Bit Number</u>
Hard cover	MCR 48
Short warning	MCR 50
Exchange request	MCR 53
External interrupt	MCR 56
System interval timer	MCR 59
Soft error log	MCR 62
Process interval timer	UCR 51

Each of the catchable conditions has a catcher rank FF before the MCR/UCR. The catcher rank is enabled by a MISC=13 or a trap microtrap generated while CPU is in job mode. Enabling the catcher rank blocks the input to the MCR/UCR. If a catchable condition occurs while the catcher rank is enabled, the respective catcher rank FF stays set until the catcher rank is disabled. Any condition caught in the catcher rank upon disabling is strobed into the MCR/UCR. The catcher rank acts only as a 50-ns delay when it is not enabled. An instruction exit disables the catcher rank.

The catchable enable FF (not to be confused with the catcher rank enable FF) controls the same group of MCR/UCR bits after the MCR/UCR. When the catchable enable FF is not set, a catchable condition set in the MCR/UCR does not cause a microtrap. However, when the catchable enable FF is set, the MCR/UCR bit remains set and causes a microtrap.

A MISC=6 or an instruction exit sets the catchable enable FF. It is cleared by specifying the PNR field or extended MISC=A, or by one of the following microtraps: parity errors, floating point exception, debug, CYBER address out of range, MAP no-hit, trap, exchange, or halt.

Non-Catchable Conditions

Any MCR/UCR bits that are not catchable conditions are referred to as non-catchable conditions. When the non-catchable enable FF is not set, a non-catchable condition set in the MCR/UCR does not cause a microtrap. However, when the noncatchable enable FF is set, the MCR/UCR bit remains set and causes a microtrap.

The non-catchable enable FF is set by a MISC=7 or an instruction exit. It is cleared by extended MISC=A or one of the following microtraps: parity errors, floating point exception, debug, CYBER address out of range, MAP no-hit, trap, exchange, or halt.

UCR bits 50, 54, 58, 59, 60 and MCR bit 60 (only during Ring number=0 and loading A registers) require both the catchable enable FF and non-catchable enable FF to set to generate a microtrap.

Microtraps MCR/UCR

Monitor Condition Register

The monitor condition register (MCR) shown on RF 2.7, is a process state, hard register located on 8TJ0, logic chassis location CP014. It can be read into the leftmost 16 bits of the AD register by specifying AD.SELECT PSR and reading register file location 05₁₆.

The MCR is written parallel to register file location 05₁₆ when the leftmost 16 bits of register file location 05₁₆ are written. The write occurs at the normal write time for the micrand, that is, 50 ns after the primitive time. The MCR bits set under the following conditions:

<u>Bit</u>	<u>Description</u>
50	Short Warning signal from UQ pak. Sets any time during a micrand.
51	Instruction Specification Error bit descriptor length greater than 63 (decimal) and MISC = 8_{16} . Sets during the second 50 ns of the primitive time of the micrand which specifies MISC = 16.
52	Address Specification Error signal from UX pak. Sets 200 ns after the primitive time of the micrand which sends the address to the MAP.
53	Exchange Interrupt signal from LDS001 pak. Sets any time during a micrand.
54	ACCESS VIOLATION <ul style="list-style-type: none"> • Signal from UX pak. Sets 200 ns after the primitive time of the micrand which sends the address to the MAP. • MAP ring number or P register ring number greater than R3 (where R3 is AD bits 12 to 15) and MISC = $1D_{16}$. Sets during the second 50 ns of the primitive time of the micrand which specifies MISC 16.
56	External Interrupt signal from LDC0 pak. Sets any time during a micrand.
59	System Interval Timer (SIT = 0) signal from LDF0 pak. Sets any time during a micrand.
60	<ul style="list-style-type: none"> • Selected Ring Number = 0 from TD pak and MISC = $1E_{16}$. Sets 100 ns after the primitive time of the micrand starts which specifies MISC = $1E_{16}$. • Code Base Pointer R3 = 0 from UX pak.
61	Outward Call Inward Return <ul style="list-style-type: none"> • Outward Call signal from UX pak. Sets 200 ns after the primitive time of the micrand which sends the address to the MAP. • AD ring number (bits 16 to 19) greater than or equal to saved AD ring number (ring field = save) and MISC = $1A_{16}$. Sets during the second 50 ns of the primitive time of the micrand which specifies MISC = $1A_{16}$.
62	Soft Error <ul style="list-style-type: none"> • Corrected CM error (from response code). Sets any time during a micrand.

<u>Bit</u>	<u>Description</u>
	<ul style="list-style-type: none"> • Retry enable FF and trap signal (retryable parity error) from TM pak. Sets during the second 50 ns of the first micrand of the retried instruction. • Any bit set in MAP CEL.

User Condition Register

The user condition register (UCR) is a process state register (RF 2.7). It can be read into the leftmost 16 bits of the AD register by specifying AD select PSR and reading register file location 04₁₆.

The UCR is written parallel to register file location 04₁₆. When the leftmost 16 bits of register file location 04₁₆ are written, the UCR is also written. The write occurs at the normal write time for the micrand, that is, 50 ns after the primitive time. The UCR bits set under the following conditions:

<u>Bit</u>	<u>Description</u>
51	Process Interval Timer Process Interval Timer = 0 signal. Can set at any time during a micrand.
52	Inter-Ring Pop AD ring number not equal to P ring number. Signal and MISC code = 1F. Sets during the second 50 ns of the primitive time of the micrand which specifies MISC code = 1F ₁₆ .
57	Arithmetic Overflow <ul style="list-style-type: none"> • 64-bit B adder overflow signal requires MISC CODE = 04₁₆. Sets at the end of the primitive time. • BDP Overflow signal and extended MISC code = 3₁₆. Sets during the second 50 ns of the primitive time of the micrand which specifies extended MISC code = 3₁₆.
58	Exponent Overflow signal from TC pak. Sets during the second 50 ns after the end of the primitive time of the micrand which caused the overflow.
59	Exponent Underflow signal from TC pak. Sets during the second 50 ns after the end of the primitive time of the micrand which caused the underflow.
61	Floating Point Indefinite signal from 1DE001 pak. Sets at the end of the primitive time of the micrand.

<u>Bit</u>	<u>Description</u>
62	Arithmetic Loss of Significance BDP overflow signal and extended MISC = 2. Sets during the second 50 ns of the primitive time of the micrand which specifies external MISC = 2.
63	BDP Invalid signal from TV pak. Sets any time between 150 ns after the beginning and 100 ns after the end of the primitive time of the micrand which caused the invalid signal.

Trap, Exchange, and Halt Microtraps

When a bit is set in the MCR or UCR, either a trap interrupt, exchange interrupt, halt interrupt, or no interrupt (stack) generate depending on the corresponding mask register, trap enable FF, trap enabled delay, and job/monitor mode FF. Either the catchable enable or non-catchable enable FF, depending on which MCR or UCR bit is set, must be set to generate one of the above-mentioned interrupts. The trap, exchange, or halt interrupt FFs set 50 ns after the MCR or UCR bit is set.

Trap Microtrap

When the trap microtrap FF is set, the microtrap sequence to location 160₁₆ occurs.

Exchange Microtrap

When the exchange microtrap FF is set, the microtrap sequence to location 162₁₆ occurs.

Halt Microtrap

When the halt microtrap FF is set, the microtrap sequence to control store location 164₁₆ will occur.

Timing

The MCR and UCR bits can be grouped as follows according to the timing relationship of the bit setting and the primitive time of the micrand which causes the bit to set:

The catchable conditions are asynchronous to the micrand. Fifty ns after the MCR or UCR bit is set, the microtrap FF (trap, exchange, or halt) sets, if the catchable enable FF is set, and blocks P register increment and RF write.

The second group includes MCR bits 51, 54 (only for MAP or P ring number greater than or equal to saved AD ring number A MISC code = 1A) and UCR bits 52, 57, and 62. These bits set in the MCR or UCR 50 ns after the primitive time starts. UCR bit 57 for B adder overflow actually sets 100 ns after the start of primitive time, but it blocks the same P register increment and RF write.

The third group includes MCR bit 60 and UCR bit 61. MCR bit 60 (only for selected ring number = 0A, MISC code = 1E) sets 100 ns after the start of the primitive time. UCR bit 61 sets at the end of the primitive time. For 100-ns micrands, both bits set at the same time. For longer micrands, UCR bit 61 maintains the same relationship with block RF write and P register increment. MCR bit 60 is like the second group regarding blocking P register increment and RF file.

The fourth group includes MCR bit 62 (for retryable parity error only) and UCR bits 58, 59, and 63. MCR bit 62 sets 50 ns after the end of the primitive time of the micrand which specifies instruction exit. The UCR bits set 50 ns after the end of the primitive time of the micrand which causes the condition. This group includes all UCR and MCR bits when they are written from WD.

The fifth group includes MCR bit 52, 54, 60, and 61. These conditions are received from MAP. The MCR bits set 300 ns after the start of the primitive time of the micrand which sends the address to MAP.

Only the UCR/MCR conditions in the second group can block the P register increment and RF write for the same micrand that generated the condition. Therefore, these are the only conditions that should be generated in the micrand which specifies instruction exit.

Debug Microtrap Operation

A debug microtrap occurs when all of the following conditions are met:

- Micrand specifies the DEBUG field.
- User mask bit 8 set.
- Trap enable FF set.
- Trap delay FF clear.
- At least one bit in the debug condition register is set.
- A corresponding bit in the debug mask register is set.
- The end of list seen flag is clear.

The RF write and P register increment are blocked for the micrand which causes all of the above conditions to be met. The block occurs only when the micrand is not sending an address to MAP.

During a format D micrand when MISC = D or MISC = 1A are not specified, the debug conditions for that micrand replaces AD bits 0 through 7 going into ARVI 2. The debug condition bits line up with the debug code bits in the debug list entry.

Processor Fault Status (PFS) Error Microtrap

If a parity error occurs in the machine during the execution of micrands, the CPU interrupts the normal sequence of micrands. The following categories of parity errors interrupt the CPU:

Category One

Category One PFS errors result in changing data in memory possibly well after the P register of that instruction has been incremented. These errors are fatal and consequently uncorrectable. The detected uncorrectable error in the MCR sets and the normal interrupt occurs.

Category Two

Category Two errors could result in the change of location contents in memory or RF and can affect the current instruction. The processor determines by microcode whether or not this error is retryable. The point of no return (PNR) is that point in time during the instruction's execution before which PFS errors can be retried without causing destructive writes. After the PNR, errors occurring in CPU are handled like Category One PFS errors.

If retry has been attempted once unsuccessfully, a Category Two parity error is handled like a Category One error. If retry is successful, the soft error log updates in the status summary register and in bit 62 of the monitor condition register.

Floating Point Microtrap

This microtrap is described later in this section.

Map No Hit Microtrap

This microtrap occurs when the MAP miss signal is active. See section V-5, MAP.

CYBER Address Out of Range Microtrap (60-Bit Mode Only)

This microtrap occurs when the FLC fault signal is active from MAP. (See section V-5, MAP.)

MISCELLANEOUS EXECUTION CONTROL

RETRY ENABLE

The CPU retries a failing instruction when bits 24 and 30 of the DEC register are 1's and if the PFS error does not corrupt the original data from the instruction.

RETRY-RELATED FLIP-FLOP

The flip-flops which implement the retry feature in CPU are as follows:

Retry in Progress

This flip-flop sets on a retry trap and clears on a successful retry. It sets the soft error bit in MCR.

Parity Trap in Progress

This flip-flop sets after a parity trap and clears when the instruction is retried. Any parity error occurring when this flip-flop is set is a fatal error.

Retry Enable Flip-Flop

This flip-flop allows parity error to trap to the retry routine if it is possible to retry the failing instruction. A mechanism in the CPU causes 50-ns errors in CPU, thereby enabling the checking of the retry mechanism. This feature is available only in job mode after the retry interval timer is loaded to a nonzero value. The timer decrements every 50 ns. When the timer reaches zero, the value of the processor test mode register dictates which error will be induced in the CPU. By controlling where and when errors are induced, it is possible to predict the nature of the error and verify the proper operation of the retry-related flip-flops.

EXECUTION UNIT ADVANCE PIPE

Any micrand full from CM gives control to the execution units which reply with either a restart clear or the signal EUAP (execution unit advance pipe) shown on PIP 2.2.

PRIMITIVE FULL (PFF)

After a restart clear, the primitive full (PFF) flip-flop (PIP 2.1) rests in a clear state until CM issues a micrand full to the execution units. An emit 64 is an unexecutable micrand which therefore clears the PFF. No primitive operations can occur.

The CSU and MAP data communication signals which normally occur on the second 50 ns of a micrand are forced to reset to zero or restart/clear when the primitive is not full.

EXIT CONDITION MET

This signal indicates that the execution units are either not busy or are ready to take and execute another primitive.

PARITY

The main 64-bit data paths in CPU have parity protection with parity generated prior to entering the WD mux, and retained through WD, the RF, the AD and BD mux, and the AD and BD registers. Parity is checked at AD and BD.

The execution units accept data before the arithmetic is done on the operands. The parity is checked and subsequently dropped. On the output of the execution units, parity is regenerated.

PRIMITIVE FORMATION UNIT

This unit provides facilities for the expansion, selection, and rationalization of instruction originating data for subsequent use within the execution units under microcode control.

The information in the primitive register provides the following:

- A common control to the execution units. Each unit decodes the control fields according to its own format and operates on the common input data from the register file.
- Register file addresses to read source data for computation and write result data.
- Execution unit selection to gate the output of one of the execution units into register file.
- Special controls to read processor state registers held in discrete flip-flops.
- Control fields to MAP and instruction/micrand pipelines.
- A copy of the program address corresponding to the instruction being executed.
- A copy of data and address fields from the instruction being executed.

The information remains in the primitive register for a minimum of 100 ns (two clock cycles).

Micrands which control long computations or which reference MAP and CM may remain in the primitive register for many clock cycles. The pipelines feeding it simply "back up". A micrand exit control field determines the length of time a micrand is in the primitive register. This field selects EXIT on a discrete event (such as data returning from central memory, or by a cycle count field). The MAC provides the external interface to the MCU, initiation of micrand sequences from a given deadstart address, and monitoring and diagnosis of the machine.

EXECUTION UNIT OPERATIONS

MULTIPLY/DIVIDE OPERATION

Multiply (General Information)

The CPU performs multiplications for integer, floating point single-precision and double-precision operands in 60- and 64-bit modes. Numeric move, decimal multiply, and decimal divide instructions use multiply where conversion between binary coded decimal operations is required.

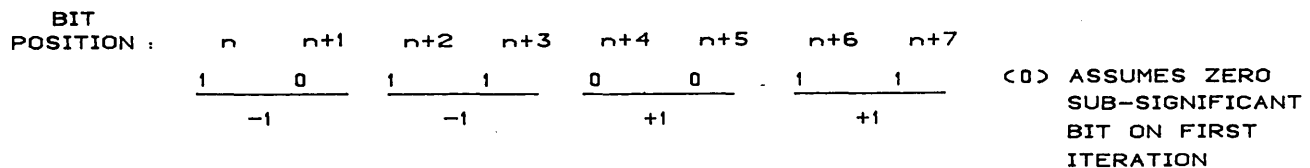
The multiply is iterative with a basic period of 100 ns (one micrand). Each iteration involves the multiplication of 2 bits of multiplier by 64 bits of multiplicand, and the accumulation of the partial product. Two bits of partial product are formed fully per micrand iteration.

Microcode drives the multiply algorithm. Each format H micrand (representing one iteration) selects the part of the partial result to be used as the final result and the number of iterations (micrands). During format H micrands, hardware controls the SBD mux and the B adder functions.

Multiply Algorithm

Booth's algorithm is used for all multiplies. This algorithm involves the separation of the multiplier into groups of 2 bits. These 2 bits are recoded to generate a set of multiples of the multiplicand. The multiples are then added to the partial product formed up to this iteration. When the adds have been completed, the whole partial product is shifted right 2 places and the next group of 2 multiplier bits is used. The shift maintains the correct positional significance of the partial product with respect to the multiples added. The value of 2 is two and the multiples of the multiplicand, determined by the recoding, are +2, +1, +0, -0, -1, and -2. The multiply operation procedures are as follows:

1. Separate from the left into 2-bit groups as shown.
2. Scan from the left each 2-bit group shown in figure V-3-7, but overlap the most significant bit of the group immediately on the right. This ensures that a string of 1's or 0's occurs only once. Note that the 2 of the previous multiplier group (two bits) is overlapped. Here 2 is assumed to be zero. This bit is also effected by the subfunction $f2 = 1$ on the first iteration.
3. Select recodes according to table V-3-1.
4. Guarantee a correctly signed result. The most significant bit (bit 0) of the multiplier should be a zero. If not, an extra iteration is required for floating point multiplies only.



C0232

Figure V-3-7. Scan Technique of the Multiplier

TABLE V-3-1. MULTIPLE SELECTION

Bit n	Bit n+1	Overlap Bit n+2	Multiple Selected	Adder Function	SBD Mux
0	0	0	+0	A+0	-
0	0	1	+1	A+B	Straight
0	1	0	+1	A+B	Straight
0	1	1	+2	A+B	Left one
1	0	0	-2	A-B	Left one
1	0	1	-1	A-B	Straight
1	1	0	-1	A-B	Straight
1	1	1	+0	A+0	-

Number of Iterations

Because two bits of multiplier are used during each iteration, the number of iterations for each instruction type depends on the 32-, 48-, or 64-bit word length.

64-bit multiply	33 iterations
48-bit multiply	25 iterations
32-bit multiply	17 iterations

The 48-bit multiply requires 25 iterations because the operands are represented in sign and magnitude, and bit 16 is always one when normalized. The floating point multiply operands may or may not be normalized. However, because it can be normalized, the setting of the most significant bit would cause a negative recode as indicated by figure V-3-8. This makes the final result negative and in error. To correct this, an extra iteration is required, with two zeros assumed on the left. This recodes a+1 recode and corrects the sign of the result.

6AE X 0E3 = ?

```

  6AE
  0E3
  ---
 140A
 5D84
  ---
 SEC4A

```

MULTIPLICAND = 6AE = 0 1 1 0 1 0 1 0 1 1 1 0 IN BINARY
 MULTIPLIER = 0E3 = 0 0 0 0 1 1 1 0 0 0 1 1 IN BINARY

1. RECODE	0 0	0 0	1 1	1 0	0 0	1 1	(0 ASSUMED)
2. GROUP	---	---	---	---	---	---	
3. RECODE	+0	+1	+0	-2	+1	-1	

1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 0 1 0 0 1 0	
0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 1 1 1 0	+1
1 1 1 1 1 1 1 1 0 0 1 0 1 0 1 0 0 1 0 0	-2
0 0	+0
0 0 0 0 0 1 1 0 1 0 1 0 1 1 1 0	+1
0 0 0 0 0 0 0 0 0 0 0 0 0 0	+0
<hr/>	
0 0 0 0 0 1 0 1 1 1 1 0 1 1 0 0 0 1 0 0 1 0 1 0	BINARY PRODUCT
<hr/>	
0 5 E C 4 A	HEX PRODUCT

C0233

NOTE

Each multiple is aligned by a 2n-bit shift left to make the significance of the multiplier recode which determined that input. The sign bit is extended to fill the adder tree.

Figure V-3-8. Example of Binary Multiply Using a Multiplier Recoding Algorithm

Basic Micrand Flow

Micrand 1 Multiplier to BD2 register

Micrand 2 Multiplier (BD2 register) to ER register

 Multiplicand to BD2 register

 Zeros to AD2 register (partial product)

Micrand 3 (AND subsequent multiply iteration micrands)

The three recode bits (ER register bits 62, 63, 64) control selection of SBD mux [MCAND straight (X1) or MCAND left 1 (X2)] and the B adder function (A+B, A-B, A+0).

The ER mux selects ER register right two. The formed 2 bits from B adder are loaded, via ER mux, into ER register bits 0 and 1. ER register bits 63 and 64 are shifted off. The effect is that the multiplier, as it is used (two bits at a time), is shifted off and the lower multiply result is assembled into ER register from the left (two bits at a time).

Divide Operation (General Information)

The CPU performs integer and floating point divide instructions in 60- and 64-bit mode. The divide algorithm is a modified non-restoring division where one quotient bit forms per iteration. Each iteration is one micrand (100 ns).

Number of Iterations

- Integer divides in 64-bit mode have a variable iteration count that is dependent upon the normalization count of the divisor.

Iteration Count (number of micrands) = NC+1

where NC = normalization count of divisor

- Floating point divides in 60- and 64-bit mode require 48 iterations.

General Divide Algorithm

- Make the dividend and the divisor positive.
- Normalize the divisor (not on floating divide).
- Put the dividend in AD2 register zero in EA normalized divisor in BD2 register function = +AD-BD

- If B adder is positive (0), record 1 in rightmost bit of ER B adder (Left 1) to AD2 register B adder function = +AD-BD.

If B adder is negative (1), record 0 in rightmost bit of ER B adder (Left 1) AD B adder func = AD+BD.

- Iteration count = NC+1 where NC is the normalization count.

Floating Point Exceptions

Abnormal floating point numbers produce a floating point exception condition (ESC = 5) in association with a floating point exception branch index that is unique for each possible floating point condition. This unique index causes, via microcode, a canned result to be written to the result register.

Early Exceptions Canned Results

The floating point function field (FUNC) selects the correct floating point exception matrix and the encoded values of the exponents select the index from the matrix as follows:

<u>Exponent Value</u>	<u>Encoded Value</u>
+ Normal	000
- Normal	001
+ Indefinite	010
- Indefinite	011
+ Infinite	100
- Infinite	101
+ Zero	110
- Zero	111

If during 64-bit mode, either exponent add or subtract equals zero, an early exception (ECS = 5) is set and a branch to a fix-up routine occurs. For 60-bit mode add and subtract, zero exponents do not produce an exception.

DEBUG OPERATION

The debug list is scanned after instruction fetch and before instruction execution if the following occurs:

- Traps are enabled.
- Bit 56 in the user mask register is set.
- One or more bits in the debug mask register apply to the instruction to be executed.
- The end of list seen flag in the debug mask register is clear.

The performance degradation begins as soon as debug is enabled. That is, traps are enabled and bit 56 in the user mask register is set. The debug list is scanned by reading the first word from the debug list in central memory at the PVA specified by the contents of the debug list pointer register.

After the first word of the debug list is read, each successive word from the debug list is read by incrementing by one the 6-bit word-index field contained in the debug index register and referencing the debug list at the PVA specified by the initial contents of the debug list pointer register. This register is modified in its rightmost 32-bit positions by the addition of the zero-extended, 6-bit word-index from the debug index register.

The debug bit in the user condition register is set, the execution of the instruction is inhibited, and a trap interrupt occurs when:

- One or more bits in the debug mask register are set and are equal to one or more of the corresponding leftmost 5 bits of the debug code in the first word of a double word entry in the debug list.
- One or more of the PVAs associated with the instructions execution are within the address range defined by the corresponding double word entry from the debug list.

When the end of list bit in debug code is set or the 32nd double word entry from debug list has been scanned, the end of list seen flag in the debug mask register sets. The second word of the double word debug list entry which causes a trap interrupt is identifiable by the 6-bit value of the debug index register and the PVA contained in the debug list pointer register.

The debug index and flags provide the means for properly initiating, resuming and terminating debug scan operations, particularly when an instruction's execution has been inhibited by one or more interrupts. The debug flags are end of list seen and scan in progress, bits 9 and 10 of word 36 in the exchange package.

These interrupts may be either trap or exchange interrupts. Exchange interrupts cause storage of the flags and debug index register in the exchange package to allow, for example, resumption of a partially completed scan of a debug entry list.

On a trap interrupt the processor retains the flags and index register to allow proper completion of the debug scan upon return from the trap interrupt. The alteration (other than clearing the flags and index to reinitiate the debug scan) of the debug index, debug flags, or UM56 during the suspension of a debug operation because of an interrupt, causes undefined debug operation on the first instruction executed after resuming debug. If traps are enabled during the processing of a debug trap interrupt, software must not re-enable debug or the integrity of the interrupted debug scan is lost.

The scanning of the debug list before instruction execution includes all instruction results except the following which may occur before the debug scan is complete:

- Setting page used bit either explicitly as in test page table (OP code 16) or implicitly as with any instruction.
- Setting of condition register bits.
- Rounding up of A0 on CALL instruction (opcode B0 or B5).
- Storing current environment into stack frame save area (SFSA) on CALL instructions. Note that the debug trap will also round up A0 and store the environment into the SFSA.

The exception testing and debug scan are not constrained to occur in any given sequence relative to each other. There is only one trap interrupt for each double word debug entry having a match or matches. (Two or more matches within the same entry produces only one trap.) The traps resulting from execution testing may occur concurrent with a debug trap (several bits set in MCR and/or UCR) or separately, either before or after the debug scan.

For the purpose of establishing central memory access (CMA) validation, each CMA performed for the purpose of reading a word from the debug list as part of a debug scan operation is a read type access.

DEBUG MICROTRAP

A debug microtrap causes the microtrap to sequence to control store location 166₁₆. UCR bit 8 may be set under micrand control as a result of the debug list scan. The debug list scan routine begins at control store location 166₁₆ in the CPU instruction set microcode. The comparison of the debug condition register, debug mask register, user mask register, bit 8, trap enable FFs, and end of list seen flag causes a debug interrupt.

The debug condition register has four bits which are set from a decode of the map function field according to table V-3-2. The fifth debug condition register bit, instruction fetch, is always set. The debug condition register strobe is enabled by format D and not MISC = D (allow AD left 16 bits to WD mux) or MISC = 1A (check for saved AD ring number greater than current AD ring number). An instruction exit clears the debug condition register.

TABLE V-3-2. DEBUG CONDITION REGISTER

Map	Function	Debug Condition
0	RMA read	-
1	RMA write	-
2	Purge buffer per K	-
3	Ibranch jump	Branch
4	Ibranch instruction	Branch
5	Ibranch call	Call
6	Ibranch return	Branch
7	Ibranch exchange	-
8	Read multiply RMA	-
9	Load FLC set 170	-
A	Load FLC Clear 170	-
B	Load PSM	-
C	Load associate file	-
D	Load real file	-
E	Load validity file	-
F	Load F latch	-
10	Read multiply NM	Read
11	Read request NM	Read
12	Write request NM	Write
13	Translate no value	Read
14	Read request BS	Read
15	Read request M8	Read
16	Write request M8	Write
17	Write multiply NM	Write
18	Load RNI P	Read, Write
19	Read set PVA	Write
1A	Read clear PVA	Write
1B	Read set RMA	-
1C	Read clear RMA	-
1D	Read exchange register	-
1E	Read FRC	-
1F	Write multiply RMA	-

SECTION V-4

READ NEXT INSTRUCTION

=====

This section describes the detailed hardware operations of the read next instruction (RNI) unit in the CPU. The RNI unit fetches central memory (CM) words containing linear instruction sequences, buffers these words, and selects individual instructions for execution. The addressing hardware can also stream operations on behalf of the execution unit.

HARDWARE OVERVIEW

The hardware (RNI 1.0.) includes the following registers and muxes:

- P0 register
- P1 register
- I multiplexer (I mux)
- RNI address register/incrementer
- P register/incrementer

P0 REGISTER

The P0 register (RNI 2.0) holds a 64-bit word which contains executable instructions. Instructions 16 or 32 bits long (15 or 30 bits in 60-bit mode) are selected from page offset (P0) via the I mux. Data is input to P0 from the P1 register, and a full/empty control bit determines if instruction data is in P0. If P1 is full and P0 is empty, P0 is set full. P0 remains full until the select code (ISEL) to the I mux increments beyond the end of P0. Then P0 is set empty and is ready to accept new data from P1. P0 is an unclocked latch which, when P0 is empty, is held open to allow data to input from P1.

In 64-bit mode, P0 passes the data unaltered; in 60-bit mode, P0 selectively shifts portions of the data one position to the right. A parity correction scheme maintains correct byte parity.

P1 REGISTER

The P1 register (RNI 2.0) holds a 64-bit word which contains executable instructions. The entire contents of P1 pass in parallel to P0 when it is empty. CM inputs data to P1 under control of the returning identification (ident) bits.

P1 is set full when the ident bits are decoded indicating a valid RNI data word from CM; it is set empty when P0 is empty, indicating that data passed from P1 to P0.

In 64-bit mode, P1 passes the data unaltered; in 60-bit mode, P1 selectively shifts portions of the data two positions to the right. A parity correction scheme maintains correct byte parity.

I MULTIPLEXER (I MUX)

I mux is a 32-bit, 4-input multiplexer (RNI 2.1). Using its upper and lower portions independently, it selects the following data fields from P0, P1, and the F latch:

- P0 (0-31)
- P0 (16-47)
- P0 (32-63)
- P0 (48-63)/P1 (0-15)
- F latch (0-31)

The first four selections pass 32-bit instructions to the control store input for execution. If an instruction is only 16 bits long, the next sequential 16 bits are sent to fill out the 32-bit data path. If this causes the instruction to flow onto the P1 register (ISEL = 3), the control waits until P1 is full. However, MAP validity checking is not done on P1 if the instruction is only 16 bits long. (See Validity Checking in section V-5). The last selection is for diagnostic purposes only and is described under F Latch in this section.

Logic associated with the I mux looks at the full/empty status of the registers selected as input by I mux. This logic determines the following:

- Enables parity checking of I mux data per byte.
- Determines if an RNI full signal should be sent to the control store input hardware. This signal is sent only if the data in I mux has been stable for at least 50 ns, allowing a settling time of 100 ns for the decode path to control store address register.

The data output from I mux sometimes has its upper and lower halves reversed because the portions of I mux on data paks one and two are controlled independently of the portions on paks three and four to reduce backpanel interconnections. The inverted condition, detected by means of the lowest bit of the I mux select code (bit 62), sends a Swap signal to the instruction decode hardware where the halves of the instruction interchange. The bit selections are listed below. Note that the bits of P0 and P1 are distributed across the data paks as follows:

<u>Pak</u>	<u>Bits</u>
1	0 -7, 32-39
2	8-15, 40-47
3	16-23, 48-55
4	24-31, 56-63

The I mux select control, called ISEL (parcel count), is bits 61 and 62 of the PVA associated with the leftmost byte of the instruction selected (table V-4-1).

TABLE V-4-1. I MUX SELECT CONTROL

ISEL	Data Required	Paks 1, 2	Paks 3, 4	SWAP
0	P0 (0-31)	P0 (0-15)	P0 (16-31)	0
1	P0 (16-47)	P0 (32-47)	P0 (16-31)	1
2	P0 (32-63)	P0 (32-47)	P0 (48-63)	0
3	P0 (48-63)/P1 (0-15)*	P0 (48-63)	P1 (0-15)	1
* The symbol / indicates concatenation.				

F Latch

The F latch register (RNI 2.1) is a used only by microcode diagnostics. It allows a test sequence to inject a desired bit pattern into the control store input and instruction decode hardware as though the instruction had come from CM in the customary manner. This allows checking of the instruction decode hardware without relying on the integrity of CM, the MAP, port, or RNI sequencing hardware. The F latch is loaded under microcode control by MAP control code F, mnemonic LOAD F-L. Refer to section V-5, MAP. Loading the F latch sets the following conditions:

- Forces I mux to unconditionally select the F latch.
- Forces I mux to fill and sends a full signal to the control store input. This signal is timed by hardware to allow the F latch data through I mux to stabilize for 50 ns.
- Enables parity checking of the I mux data.

The F latch is cleared by MAP code A. Note that ISEL, and hence Swap are active when the F latch is in use.

RNI Address Register/Incrementer

The RNI address register holds the 48-bit PVA of the next word of instructions to be fetched from CM. The top 16 bits (ring, segment) are held in a fixed register on 8UX0 pak (RNI 2.2), logic chassis location CP024; the bottom 32 bits (BN) are held in an incrementer on the 1DW0 paks, logic chassis locations CP022, CP023, CP025, and CP026.

The RNI register is loaded from the MAP input address during RNI or stream initialization. It is incremented immediately following an RNI initialization and from then on follows every RNI request sent to the MAP. Stream operation is discussed later in this section.

The RNI register is incremented at word boundaries (bit 60). Within the constraints of a multiple-use data pak 1DW0, this is accomplished by providing backpanel pins for the bit 60 position and the group carries between incrementer sections. Dedicated backpanel wiring substitutes a discrete FF on a control pak for the bit 60 position of the counter as well as for that group's carry.

Parity is regenerated on the output of the RNI incrementer. No provision has been made to force bad parity in this generator because the downstream parity checkers are more conveniently checked by forcing bad parity on the output of the execution unit's AD register.

P Register/Incrementer

The P register (RNI 2.4) holds the PVA of the instruction executed at primitive time in the execution unit. The P register is physically packaged with the RNI/MAP to save hardware data paths. Bits 0 through 15 (keys) are on pak 8UX0 and are held in a static register for key/lock validity comparisons; bits 16 through 31 (ring, segment) are on pak 8UX0 and are held in a static register for convenient access by the microcode; bits 32 through 59 are on paks 1DW0, and are held in an incrementer; bits 60 through 62 are held in a register on pak 2GA0, logic chassis location CPU21. Appropriate bits of this register are taken to back-panel pins on 1DW0: dedicated backpanel wiring injects bits 60 through 62 into the register output bus.

When an instruction is presented to the decode hardware via the I mux, the I mux select bits (ISEL) and a carry-out bit are passed into a pipeline which has stages associated with S time and MHR time. For a discussion of pipe timing, see section V-3 on Execution. The P register is associated with primitive time. The data in this pipeline is used to form bits 61 and 62 of the P register (parcel count) and to increment the rest of the P register at bit 60. Note that bit 60 is held as a discrete flip-flop on the control pak.

The P register is loaded by a branch function (MAP function 3, 4, 5, 6, or 7) or by a reload function (MAP function 18). The P register is not incremented during a Flexit instruction exit. This ensures that P points to the beginning of the current instruction, not to the beginning of a descriptor.

Parity is regenerated on the output of the P register. No provision has been made to force bad parity in these generators.

HARDWARE FUNCTIONS

RNI INITIALIZATION

The RNI hardware is initialized to perform one of two basic functions: RNI and streaming. RNI initialization is performed by the MAP function codes in table V-4-2.

TABLE V-4-2. MAP FUNCTION CODES FOR INITIALIZATION

Code	Mnemonic	Description
3	IBRANCH JUMP	Jump
4	IBRANCH INST	Branch
5	IBRANCH CALL	Call
6	IBRANCH RET	Return
7	IBRANCH EXCH	Exchange

These functions appear identical to the RNI unit's functions: the differences in interpretation of the five codes are solely in the MAP. They involve key lock and ring checking. Within the RNI portion of this specification, the functions are referred to collectively as branch.

A branch causes the following sequence of operations within the RNI unit:

- Terminate the activities in progress.
- Declare P0 and P1 empty. Hence, I mux is empty.
- Declare one RNI request outstanding. The branch function in the MAP results in an RNI read request to the CSU using the RNI unit's ident code.
- Increment the RNI unit's ident code. See Ident Codes later in this section. This increment occurs before the ident passes to the CSU with the translated RMA from the branch address.
- Copy the input PVA (branch address) into the P register and the RNI register.
- Increment the RNI register on the following clock cycle to point to the next word to be requested.

The RNI unit is ready for steady state RNI operation.

STEADY STATE RNI

RNI Words and Addressing

The RNI buffer consists of two stages: P0 and P1. It is possible to have a maximum of two outstanding RNI requests to CM.

To initiate RNI requests to MAP and hence to CM, an RNI control counter is maintained. It counts the RNI words in P0, P1, and those outstanding in CM. It holds values of 0, 1, or 2. Upon initialization, it is given a value of one as described above. When its value is less than two, the hardware attempts to send RNI requests to the MAP. The address sent with these requests is the PVA held in the RNI register. When a request is accepted by the MAP (MAP empty to RNI), the RNI register and the RNI control counter are incremented.

When ISEL increments past the end of the P0 register, P0 is declared empty as described above. At the same time, the RNI control counter is decremented to indicate the loss of an RNI word.

The MAP translates the PVA associated with an RNI request to an RMA which is passed to CM with a memory read function. The only validity checking is on bit 32 (MAP function code 13). The currently active RNI ident is the ident passed to CM.

An ident accompanies data words returning from CM. If the ident matches the current RNI ident, a full to P1 is generated and is delayed by one clock cycle. Note that the ident precedes the data by one clock cycle.

The full/empty controls pass data words in P1 to P0. Data words in P0 are available to the I mux for selection and passage to the control store as instructions for decoding. When data is in the I mux, an RNI full to control store is generated. This full signal and data stable for 50 ns depends on the following conditions:

- Entire instruction resides in P0 and P0 full
- P0 and P1 both full
- Uses the F latch if F latch data stable

Instruction Decode for Length

The instruction passed to the control store and primitive decode is selected through I mux by bits 61 and 62 of the PVA (parcel count) associated with the instruction shown in figure V-4-1. The opcode portion of the instruction is taken onto the main RNI control pak 2GA0 where enough bits are decoded to determine the length of the instruction. The possibilities of instruction length are:

- 2 bytes (1-parcel)
- 4 bytes (2-parcel)
- 8 bytes (ECS, CMU in 60-bit mode)

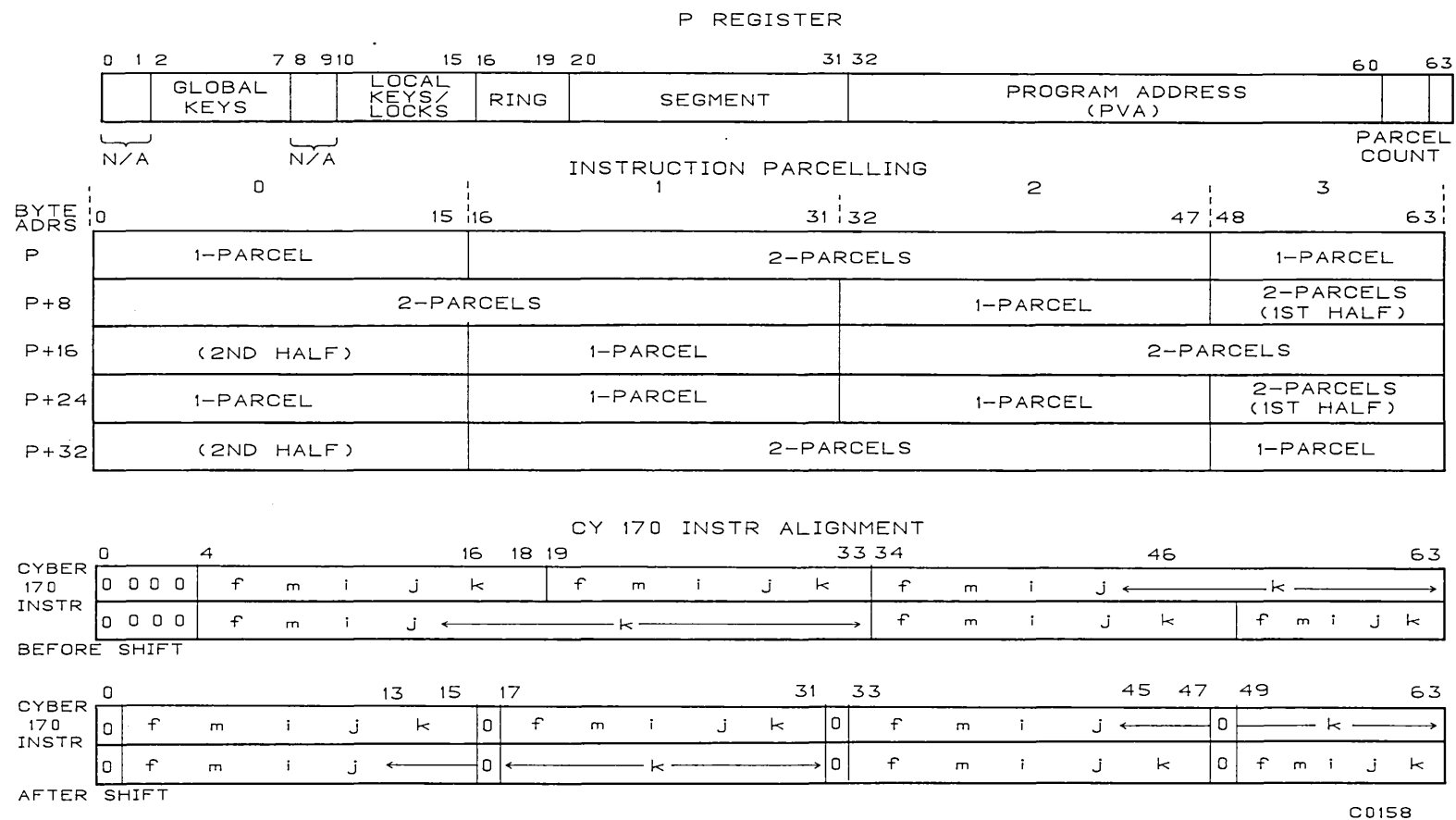


Figure V-4-1. Instruction Parcelling

Only the first two cases are differentiated because instructions, which are eight bytes long, always terminate with a branch to the next instruction to be executed. Because of this microcode convention, there is no need to provide the ability to increment the RNI selector by eight bytes. These instructions are allowed to default to an increment of four bytes for hardware convenience.

ISEL can increment by two (increment bit 62) or by four (bit 61). It consists of bits 60 through 62. Any action which produces a carry signal to increment bit 60 automatically sets P0 empty and decrements the RNI control counter. The increment of ISEL occurs when I mux is full and control store signals the RNI unit that it is taking an instruction.

BDP Instruction Lengths

The BDP instruction lengths are calculated separately from the lengths of the associated descriptors: the instructions are two bytes (7x series) or four bytes (Ex, Fx series) long while the descriptors are always four bytes long.

The procedure described above passes the instruction portion to the control store, that is, the signal, taking instruction, derives from the normal instruction exit code in the micrand. The descriptor is taken by a mechanism called the flaky exit. This function uses a MAS control field of five with strobe R1 (mnemonic FLEXIT) and creates the control signal, Taking Descriptor, which does the following:

- Forces increment of ISEL by four
- Inhibits debug pretrap

Other FLEXIT functions are described in the section on Control Store.

MAP Faults

When MAP translates an RNI address one of the following faults may occur:

- Bit 32 sets
- No hit
- FLC compare error (in 60-bit mode only)

To signal the occurrence of a fault, MAP sets a bit in the ident before passing it to CM with a dummy address. No attempt is made to differentiate the type of fault. All that is required is the knowledge of a fault. The returning ident is tested and the fault bit passes through P1 and P0 to I mux. If I mux tries to use data from P0 and/or P1 when the fault bit is in that register, a fault signal is sent to a control mux in series with the address to the control store. The fault signal causes a trap address to replace the opcode. The microprogram initiated at the trap address tests the associated P address for the reason for the fault as follows:

- Test bit 32.
- If in 60-bit mode, compare with FLC.
- Initiate an address translation for the MAP.

This technique is important in that it does not force the trap address until the control store actually tries to use the offending piece of data. The trap address is handled like an opcode, so no new timing sequences need be introduced to the machine.

Debug Pretrap

To initialize a buffer area in the scratch pad register file, a short micro-code routine must run before every instruction executed while debug is in progress. A technique similar to that for handling MAP faults accomplishes this automatically: a trap address is substituted for the opcode on alternate control store exits. This is accomplished by a flip-flop on the control pak. The flip-flop is set by a branch initialization and toggled thereafter on every normal control store exit (not FLEXIT). When set, it forces a trap address instead of the opcode. However, if a MAP fault is associated with the next instruction, a debug pretrap is not forced.

HALTING THE RNI UNIT

Master clear or the miscellaneous function KILL RNI forces the RNI unit to a quiet state (figure V-4-2). Clearing a flip-flop which normally enables RNI request to the MAP achieves quiet state. An initialization for RNI or stream operation clears this flip-flop. A timing chain which controls the branch sequence sets the flip-flop. A KILL function clears P0 and P1. This moves the full to control store.

INITIALIZATION FOR STREAM REQUESTS

Execution unit uses the RNI address register and its associated controls to generate sequential stream requests to the MAP on behalf of the execution unit. The MAP function codes shown in table V-4-3 perform stream initialization.

The RNI unit does not distinguish between the above functions: the differences in interpretation of the four codes lie solely within the MAP. When initiated, the RNI unit enters a steady-state stream mode of operation which terminates as described under Terminating Stream Operation in this section.

The following events occur within the RNI unit upon stream initialization:

- MAP input address (PVA or RMA) is loaded into the RNI register.
- The ident circuitry is forced to generate execution unit ids.
- The subsequent code is stored in a register to be used for all subsequent requests to the MAP.

- A stream mode flip-flop is set to force repeated address requests.
- A signal is sent to the execution unit to decrement the length counter if the MAP function is a read request.

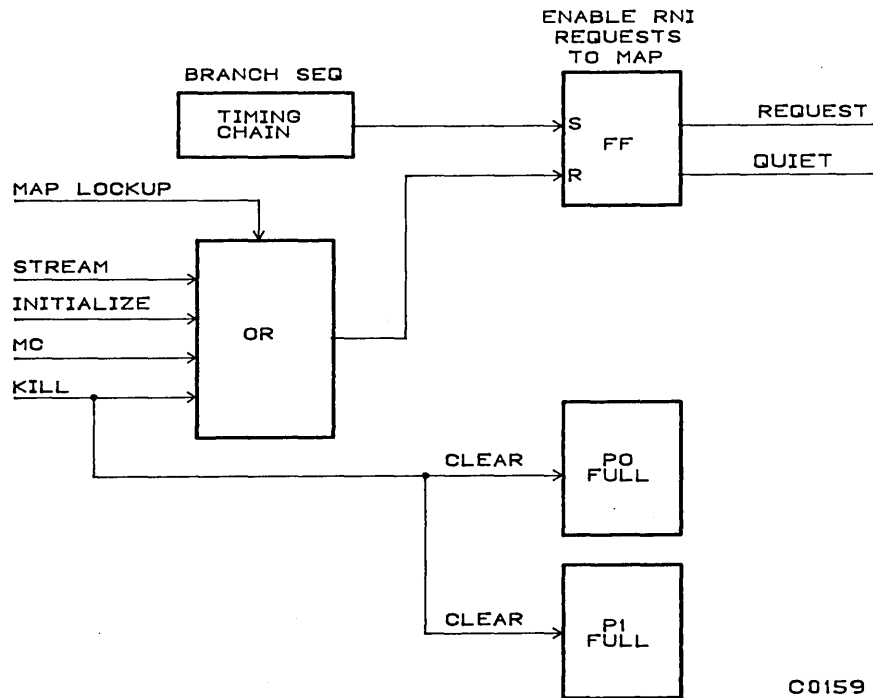


Figure V-4-2. Halting the RNI Unit

TABLE V-4-3. MAP FUNCTION CODES FOR STREAMING

Code	Mnemonic	Description	Subsequent Code
3	WRITE MULT RM	Initialize Stream, Write, RMA	1
8	READ MULT RM	Initialize Stream, Read, RMA	0
10	READ MULT NM	Initialize Stream, Read, PVA	11
17	WRITE MULT NM	Initialize Stream, Write, PVA	12

NOTE

When initialized for write operations (codes 3 and 17), the MAP can accept the first piece of write data immediately. However, it must wait 200 ns before it can accept the second item. The 100 ns of dead time is needed to initialize the RNI address counter. No interlock is provided. A stream write operation must wait 100 ns (one dummy micrand) between initiation of the stream and beginning of sending the write data.

STEADY-STATE STREAM OPERATION

The RNI unit generates a steady stream of requests to the MAP. Each request is accompanied by the PVA (or RMA) in the RNI register and by the subsequent function code. The stream rate is controlled by the rate at which the MAP accepts requests (which is ultimately controlled by the port) and, in write operations, by the data rate from the execution unit. Each request accepted by the MAP causes the RNI register to increment at bit 60 and sends a signal to the execution unit to allow decrementing of the length counter.

RESETTING RNI OPERATION AFTER STREAMING

During a stream operation, the contents of P0 and P1 remain unchanged. In addition, any RNI requests to CM that were outstanding at the time of stream initialization are loaded into P1 to continue as in normal RNI operation. To return to RNI operation, reload the RNI register. MAP function code 18 (mnemonic RNI) accomplishes this by the following:

- Reloads the RNI register with the address of the next instruction to be executed. This address is copied from the MAP input register.
- Returns ident codes to RNI values.
- Increments the RNI register according to the sum of P0 full plus P1 full so that it points to the next word requested from CM.
- Clears the stream mode flip-flop and sets the RNI mode flip-flop.

TERMINATING STREAM OPERATION

Stream operation is terminated by the following events:

- Length counter in execution unit equal to zero
- MAP lockup because of page fault or validity fault
- Kill RNI (described earlier in this section)

Each of these events clears the flip-flop which enables stream requests to the MAP. A timing chain which controls the initialization sequence sets the flip-flop.

NOTE

If length is equal to zero at initialization, no requests are sent to CM and the stream flip-flop does not set. However, access validation is performed on the address which accompanies the initialization function.

60-BIT MODE OPERATION

The facilities described in this section are intended for support of 60-bit mode execution. They are enabled by the signal RNI/MAP in 60-bit mode which is set when the FLC register is loaded (MAP function SET) and cleared when the FLC register is cleared (MAP function CLR).

60-Bit Mode Shift

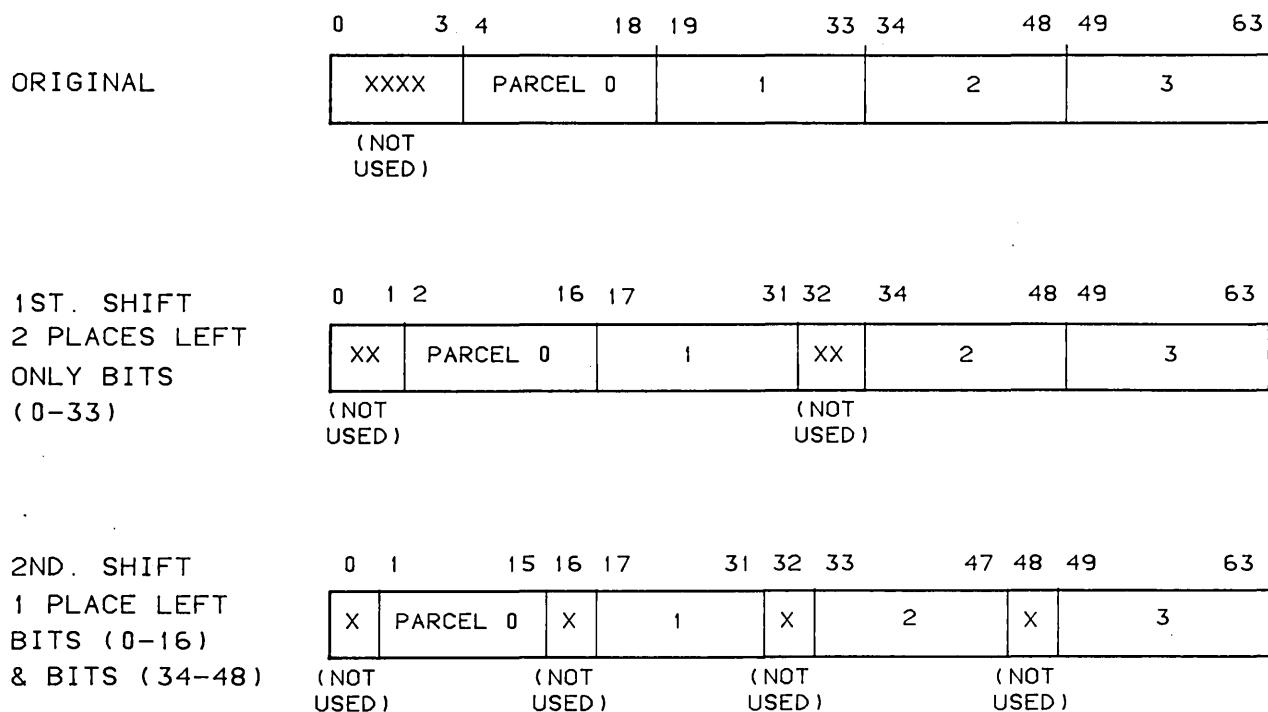
The instructions in 64-bit mode are arranged as four parcels of 16 bits within a 64-bit CM word. The instructions in 60-bit mode are arranged in four parcels of 15 bits each, right-justified within the 64-bit word, with the leftmost four bits undefined. The 60-bit mode shift in the RNI unit moves the parcels left where X is an unused bit. In the lower drawing shown in figure V-4-3, the unused bits are copies of input bits (from CM) 3, 18, 33, and 48 respectively.

The shift is accomplished using the dual inputs of the P0 and P1 registers. P1 is controlled in 60-bit mode to select the input shifted by two positions for input bits 0 through 31. This shifts bits 0 and 1 out of the register and shifts bits 32 and 33 in from the next portion of the word. The P0 is similarly controlled in 60-bit mode to provide a further shift of one position for the bits originally corresponding to input bits 0 through 15 and 32 through 47. Parity is maintained throughout the shift operation.

60-Bit Mode Length Decodes

In 60-bit mode (RNI 2.1), the instruction lengths possible are: two bytes, four bytes, and eight bytes shown in table V-4-4. All eight-byte instructions (CMU, Program Stop, ECS, etc.) conclude in P1 with an unconditional branch to the next executable instruction. Hence, the RNI unit differentiates between only two and four-byte lengths. Some of the decodes are modified for hardware conveniences.

These lengths are used for the dual purpose of incrementing the ISEL bits and detecting parcel boundary faults. Parcel boundary faults are blocked for opcodes 00 and 014 to 017 (octal).



C0197

Figure V-4-3. 60-Bit Mode Shift

TABLE V-4-4. 60-BIT MODE LENGTH DECODES

OP Codes (Octal)	True Bit Length	RNI Bit Length Decode
00-013, 02-07	mixed	30
014-017	15	15
10-37	15	15
40-463	15	15
464-467	60	15
50-52	30	30
53-57	15	15
60-62	30	30
63-67	15	15
70-72	30	30
73-77	15	15

Parcel Boundary Faults

In 60-bit mode, all instructions are length-checked to determine if they cross the boundary from P0 to P1. If they do, a parcel boundary fault signal is sent to the input of the control store. This signal forces the taking of a trap address in place of the opcode at instruction exit time. Those instructions which are legal only if they begin in parcel 0 are checked by microcode for parcel address validity.

Word Boundary Checking

60-bit mode can accept 60-bit mode exchange requests only at the completion of execution on a word boundary. ISEL bits 61 and 62 are checked constantly for a value of zero. This enables exchange traps from IOU.

MAINTENANCE FUNCTIONS

READ P REGISTER

Microcode can read the P register directly using the MAP sense lines (MAP 2.5). If there is no read MAP function from the microcode, the P register is available by default on this interface.

READ RNI REGISTER

The UTP register in the MAP tests the BN portion of the RNI register. Following initialization, the RNI unit immediately sends one request to the MAP by using the incremented address in the RNI register. This address remains in the UTP register. The procedure to test the RNI register is as follows:

1. Issue a branch function.
2. Wait for MAP all quiet.
3. Read MAP by selecting UTP register.

A similar result can be obtained by performing stream initialization with a stream length of two.

PERFORMANCE MONITORING FEATURES

The following signals are available to the performance monitoring facility. Each signal assumes a one state when the description is TRUE.

<u>Signal Description</u>	<u>Event/State</u>
RNI request to MAP	Event
Stream request to MAP	Event
I mux empty after branch	State
I mux empty, not after branch	State

IDENT CODES

The eight ident tag bits 0 through 7 (figure V-4-4) available for use by the CPU is as follows:

<u>Bit</u>	<u>Use</u>
0,1	Unused
2	RNI data when set, execution data when clear
3	Fault in RNI translation
4-7	RNI compare code

RNI REQUESTS

The RNI unit maintains a 4-bit ident code counter (RNI 2.3). The counter is initialized to all 1's on master clear, and is incremented by each branch operation. Each RNI request to the MAP is accompanied by the 4-bit code. As the ident passes through the MAP for an RNI request, the MAP sets bit 2 to indicate an RNI request and, in addition, sets bit 3 if a translation fault has occurred.

Data from central storage which is accompanied by an ident with bit 2 set causes a compare of bits 4 through 7 with the ident counter. A match causes a full to P1. Bit 3 is copied into the P1 status register to indicate a MAP fault.

BRANCH REQUESTS

The execution unit does branch requests by microcode. The MAP sets bit 2 and appends the incremented contents of the ident counter as the lower four bits of the ident code.

STREAM REQUEST

Stream requests by the RNI unit clear bit 2 so that returning data is directed to the execution unit.

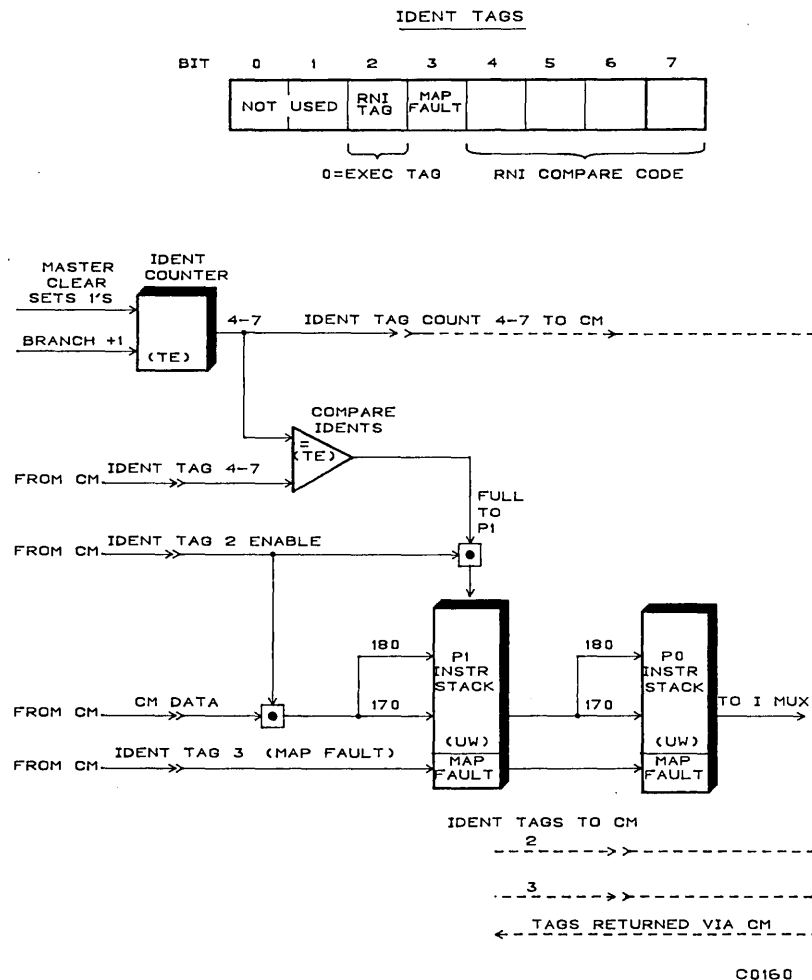


Figure V-4-4. Ident Tags

SECTION V-5

MAP

=====

The external interfaces and internal structure of the central processor's MAP are logically and physically associated with the processor's read next instruction (RNI). Refer to RNI, section V-4. The MAP unit shown in figure V-5-1 provides address translation and verification. No facilities exist to search the segment and page tables in central memory. The execution unit performs these tasks under microcode control.

The MAP accepts input addresses and control functions from either the execution or RNI unit. These addresses are loaded into the MAP registers as control parameters for subsequent MAP functions, or can be modified according to defined algorithms to produce output real memory addresses (RMA) for use by central storage. This section describes control registers (FLC, files, page size mask, etc.) and MAP translation sequences.

UNIT DESCRIPTIONS

MAP INPUT REGISTER

The MAP input register (MAP 2.0) holds the input address while the MAP operates on it. A multiplexer (mux), which accepts addresses from either the execution or RNI unit, loads this input register.

FLC REGISTER

The central memory field length register (FLC) is loaded from MAP input register. The FLC (MAP 2.6) holds a 32-bit BN value used in 60-bit mode to test for field length (core) violations. The 21-bit (60-bit mode) address extends to a 32-bit BN within execution unit as follows:

- Zero fill bits 61 through 63 in the byte offset bit positions.
- Copy FLC into bits 40 through 60.
- Zero fill bits 32 through 39.

All addresses within 60-bit mode conform to this format. Loading FLC sets a bit which sets the RNI unit into 60-bit mode and enables comparisons of all subsequent RNI process virtual address (PVA) byte numbers to FLC. The value in FLC is also compared against all execution requests when in 60-bit mode.

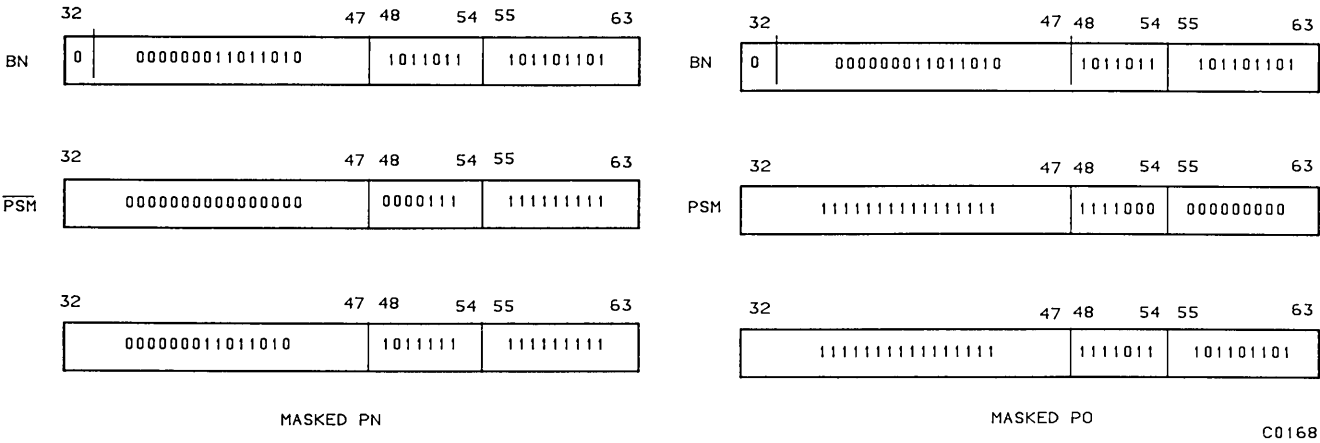
PAGE SIZE MASK REGISTER

Masking is a technique in which an unused portion of data is forced to a defined value of all 1's.

The page size mask register (PSM) is a 7-bit entity (MAP 2.0) which holds a mask of advancing 1's from the left. The PSM operates on bits 48 through 54 of the input PVA to define the boundary between page number to be translated and page offset to be copied unchanged (figure V-5-2). The PSM has 1's in those bit positions which belong to the page number.

Before loading the PSM, the execution unit's microcode extends it to fill a 32-bit register as follows:

- Bits 32 through 47 must be 1's.
- Bits 55 through 63 must be 0's.



C0168

PSM VALUES

48	54	PAGE SIZE
1 1 1 1 1 1 1		512 BYTES
1 1 1 1 1 1 0		1024 BYTES
1 1 1 1 1 0 0		2048 BYTES
1 1 1 1 0 0 0		4096 BYTES
1 1 1 0 0 0 0		8192 BYTES
1 1 0 0 0 0 0		16K BYTES
1 0 0 0 0 0 0		32K BYTES
0 0 0 0 0 0 0		64K BYTES

Figure V-5-2. Formation of Page Number and Page Offset

The TRUE and FALSE values are Ored bit-by-bit with the BN to produce the following:

Masked PO

Description

BN.OR.PSM

Page number portion is forced to 1's;
Page offset portion is unchanged.

Masked PN

BN.OR.NOT.PSM

Page offset portion is forced to 1's;
Page number portion is unchanged.

ANDing the masked PO and masked PN could reconstruct the original BN. To test for a hit the masked PN is compared with the contents of associative files. The contents of associative files are similarly masked by the microcode prior to loading the files. The masked PO inserts page offset bits into the output address. They are ANDed with the real file output to form the output RMA. The microcode ORs the real file contents with the inverse of PSM to form a masked PN prior to loading the files. A second copy of PSM required in bits 25 through 31 of the input data controls the selection of the bottom four bits of page number for use in the hash address to MAP files (see below).

ASSOCIATIVE FILES

The associative files shown in figure V-5-3 (MAP 2.0) contain segment numbers and masked PN addresses to be compared with the input PVA for match. In this section a match is called a hit.

The associative files which are accessed in a hashing manner have the following characteristics:

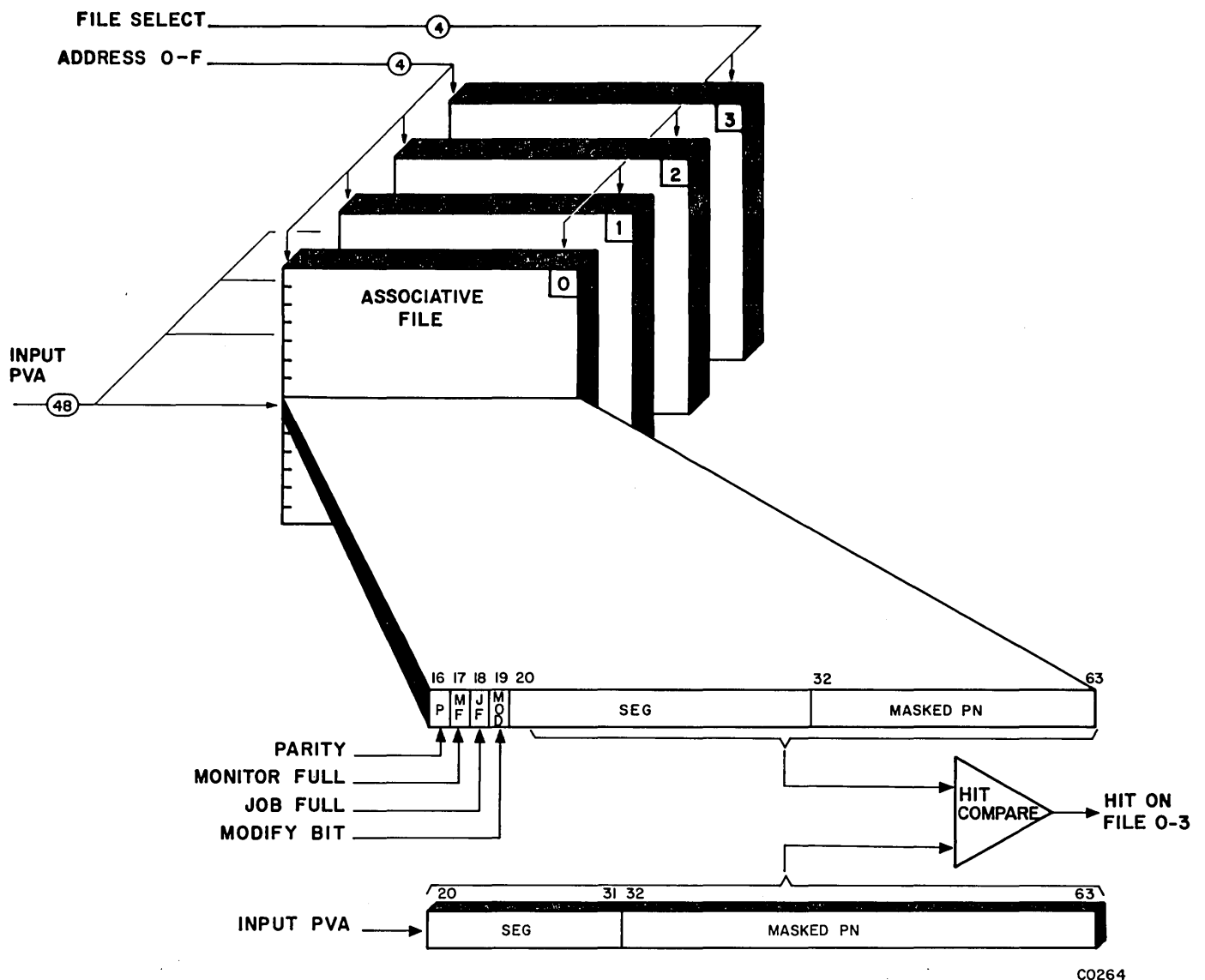
- Number of file addresses equals 16.
- Number of entries at each file address equals 4.
- Width of each entry equals 44 bits.

The hash file is accessed by a hash address constructed by exclusive ORing the least significant four bits of segment number with the least significant four bits of page number (the four bits are selected from page number per the PSM). The four entries found at the resulting file address are compared in parallel to masked PN and segment number generated from the input PVA. The comparison for each entry is enabled by control bits which define the following:

- Entry full, available in monitor mode.
- Entry full, available in job mode.
- Entry has page-modified bit set.
- No parity error detected in this entry.

RMA REGISTER

After MAP manipulation, the 32-bit RMA register (MAP 2.1) holds the address for presentation to central storage. This register is strobed at the completion of a MAP function only.



C0264

Figure V-5-3. Associative Files

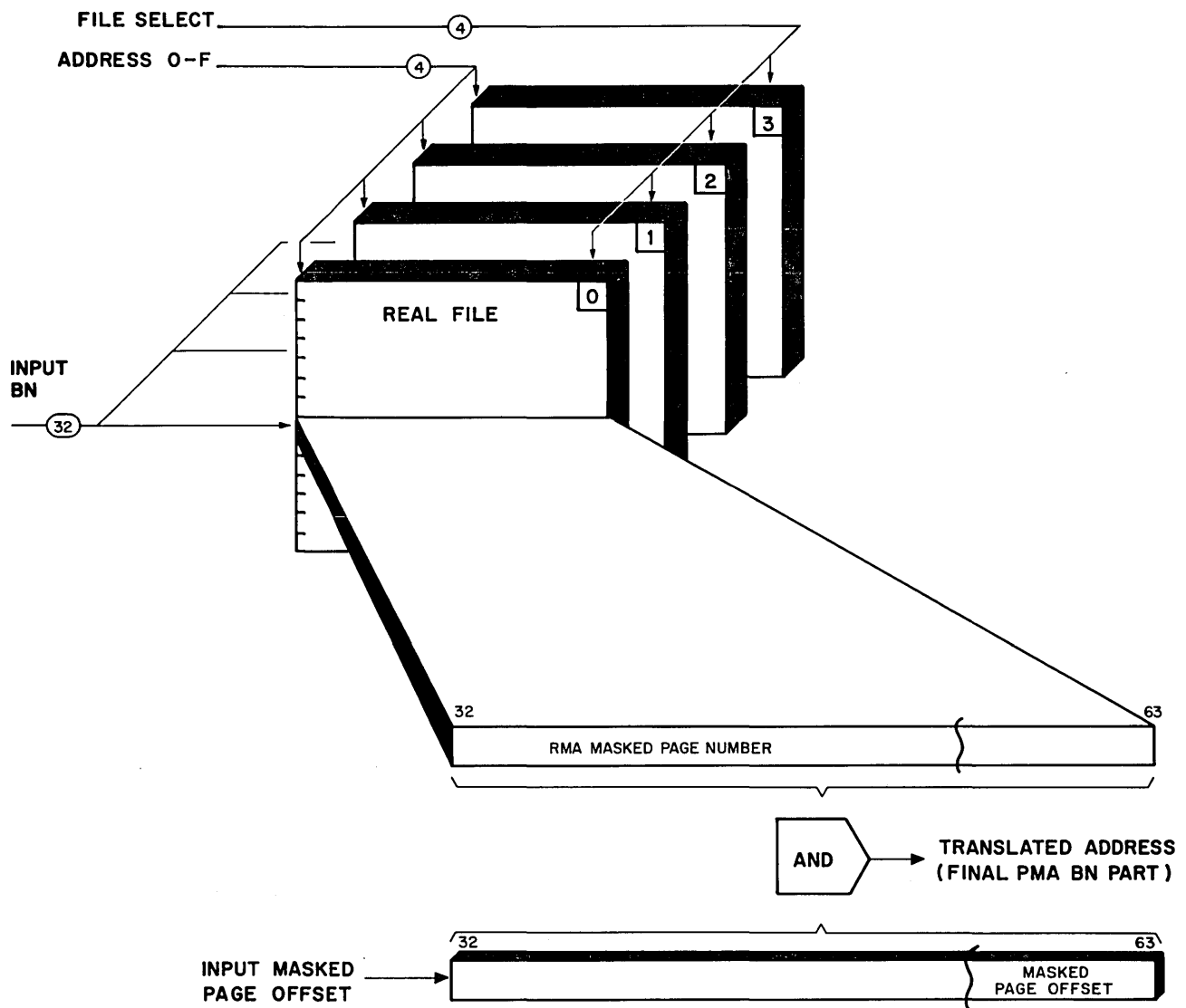
A hit on one entry causes the corresponding entry to read out of real and validity files to form the RMA and perform validity checking. A hit on more than one file sets the multiple hit error bit in the processor fault status register.

Entries in the associative file are loaded by microcode and cleared by either a purge buffers function or a request to perform a write function. The entry having a hit does not have its modified bit set.

Parity is checked when each associative file entry is used. A parity error blocks a hit and sets a bit in MAP-corrected error log.

REAL FILES

The real files shown in figure V-5-4 (MAP 2.0) contain masked page frame addresses which generate translated RMAs for central memory.



C0265

Figure V-5-4. Real File

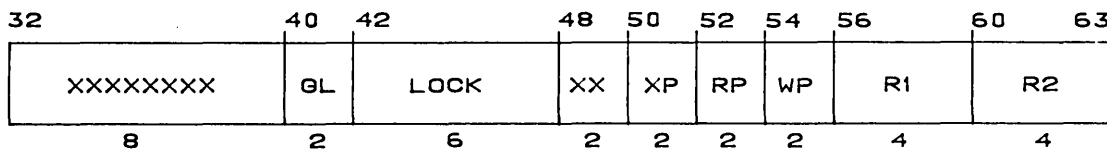
Real files have the same number and size of entries as associative files. An entry is selected from real files when the file number has a hit in the associative file. The real file value is ANDed with masked PO to form the final RMA.

Parity is checked when each real file entry is used. A parity error blocks a hit on the associative file entry and sets a bit in the MAP-corrected error log.

VALIDITY FILES

The size and operation of the validity files shown in figure V-5-5 are similar to real files except that parity is checked after the validity bits are loaded into the output holding register. Hence, an error is fatal and sets a bit in the processor fault status register. The validity memory files are shown on MLBD MAP 2.0.

Validity file data is used to check access validity. The data is formatted by microcode as follows before loading the files:



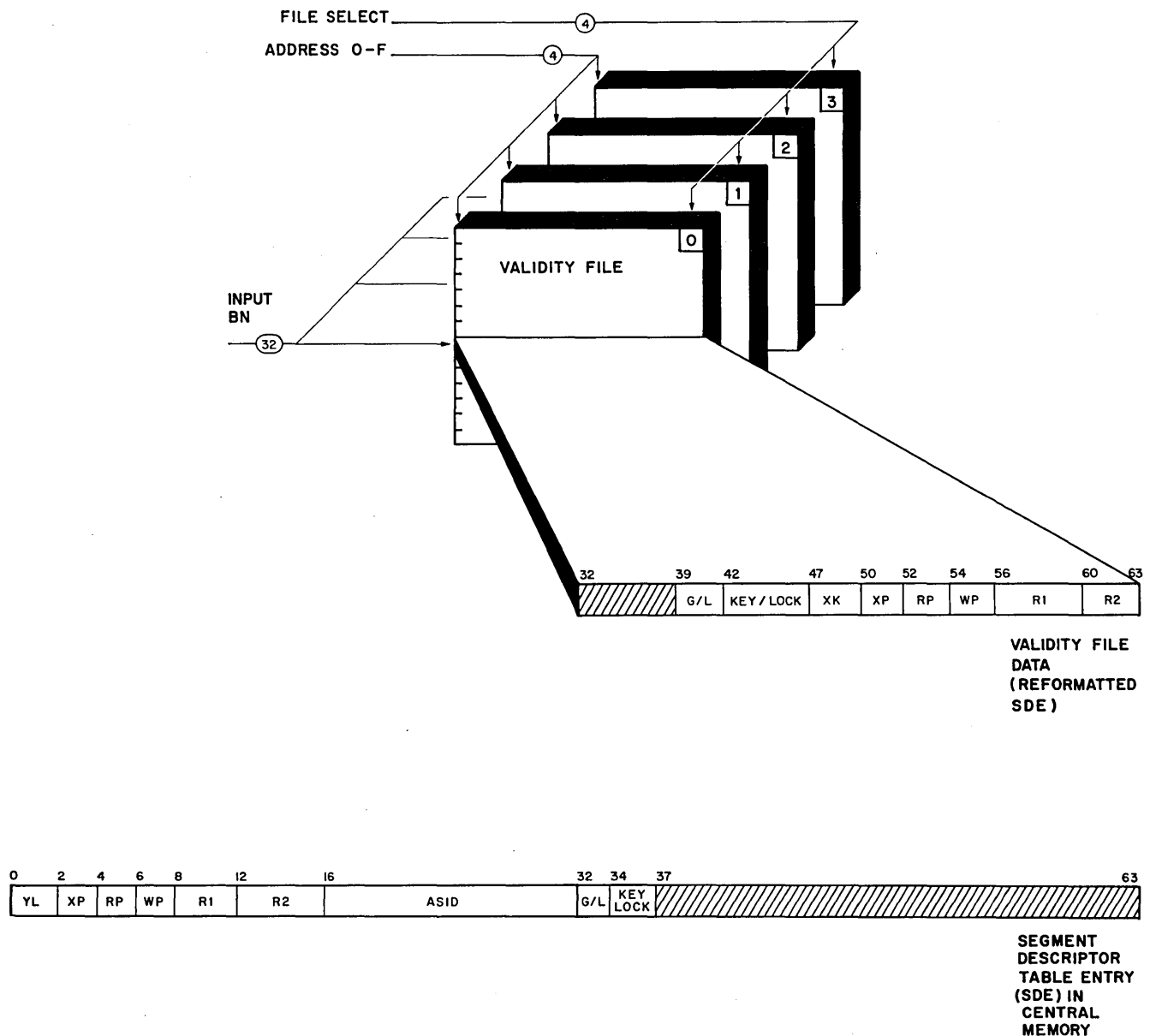
C0169

The size of each field is indicated along the bottom of the above diagram. Note that this data comes entirely from the contents of the segment descriptor in CM. The data must be reformatted before being loaded into the files. Unused bits 32 through 39, 48, and 49 of validity file have no effect on the validity checking mechanism.

UTP REGISTER

The untranslatable pointer (UTP) and RMA registers are strobed at the completion of every MAP function (MAP 2.1). The UTP register loads a copy of the input PVA (ring, segment, and BN) presented for translation. For an RMA function, the upper 16 bits of UTP load in those bit positions what the execution unit sends during the input function.

If a validity fault or a miss occurs, the RMA and UTP registers lock so that the PVA in the UTP can be read by microcode and placed in the exchange package or used to perform a search of the segment and page tables in CM. A sense condition, MAP locked, is sent to control store to test for this condition.



C0266

Figure V-5-5. Validity Files and SDE

MAP SENSE MUX

The MAP sense mux (MAP 2.5) allows the reading of all files and several registers by microcode for diagnostic purposes. If not in use, sense mux selects P register onto the output bus. The required select codes and the quantities which sense mux can read are detailed later in this section.

HARDWARE FUNCTIONS

HARDWARE SEQUENCES

All hardware sequences are initiated by requests made to MAP by execution unit (microcode) or the RNI unit. For simultaneous requests, the execution unit has priority.

HARDWARE CYCLES

In the absence of CM conflicts, MAP requires the number of clock cycles shown below to accomplish its functions. The MAP interface, as sensed by execution or RNI unit, appears empty until a request is accepted. MAP then appears full for the number of clock cycles required to complete the function. It empties again to allow strobing of a new request into the input register. For functions which specify a central memory write operation, MAP delays exiting until the data is received from execution.

<u>Function</u>	<u>Cycles</u>
Purge MAP, except Code A	17
Purge per PVA (Code A)	2
All other functions	2

CAUTION

Every write function must receive the data to complete or MAP hangs. Use master clear or MAP kill to clear.

Port Busy

The central storage unit has an input buffer rank reserved for requests from P1. When this rank is not available to accept a new address/data entry, it sends a port busy signal to MAP. This signal holds MAP data in the RMA register (MAP 2.1).

The RMA register has a full signal which delays the normal exiting of MAP function when port busy occurs. Hence, a memory conflict makes the MAP pipeline wait until the conflict is processed. This affects the number of cycles needed to perform a MAP function.

RMA Path

This path (MAP 2.1 and MAP 2.2) bypasses all translation activities and copies the BN portion of the input address directly into RMA register. This path is used in the following instances:

- MAP functions which specify that the input address is an RMA, that is, no translation required

- MAP functions which load internal MAP registers and do not translate
- Any MAP function where the environment control register bit 25 is set (RMA mode)

RMA mode does not affect the number of clock cycles required to perform a MAP function.

UTP Register Strobing

The UTP register (MAP 2.1) is strobed at the end of every MAP function and loads in a copy of the input address used during the function regardless of the type of function.

The UTP register may be read by microcode through MAP sense mux. The contents of the UTP register represent the last input address presented to MAP, either by execution unit or RNI unit. Note that if MAP is not quiet (microcode branch condition) before the UTP register is read, a changing data value during the reading could cause parity errors in execution unit.

If a MAP miss or a validity fault occurs during translation, the UTP register locks to hold in the PVA which caused the fault. The register (described later in this section) remains locked until an execution request from microcode opens it.

Validity Checking

MAP validity tests (MAP 2.6) in the clock cycle following translation. That is, during validity testing, RMA sits in RMA register and a request is sent to central storage. When a fault condition is known, RMA may already have been loaded into the port input buffer. Thus, a signal is sent to central storage which aborts previous request. The signal causes the port to change the memory request function to a read.

Fault Reporting

The reporting of a MAP miss or validity fault (MAP 2.6) depends on the initiator of the input function.

Functions Initiated by Execution

Any fault is fatal. The MAP sends appropriate fault bits directly to execution unit to be used as follows:

- FLC fault causes a trap to initiate exchange to 60-bit mode monitor, or an exchange to 64-bit mode monitor if already in 60-bit monitor (microcode routine). An FLC fault over-rides a MAP miss fault, but not a validity fault.

- MAP miss causes a trap to initiate microcode translation through central memory segment and page tables. At the successful end of this microcode routine, the files in MAP load with the translation and validation parameters.
- Validity fault sets the corresponding bit in monitor condition register.

Functions Initiated by RNI

The RNI unit generates only one MAP function when requesting RNI data.

Translate with No Validation (Function Code 13 Hex)

The following fault conditions may result from this function:

- MAP miss
- Bit 32 equals zero
- FLC fault

The occurrence of one or more of these faults sets a bit in the ident code passed to central memory. This bit is tested by RNI unit when the ident is returned by central memory. Refer to the RNI section for details.

RNI-Initiated Stream Request

For stream operations, RNI requests are treated as execution requests. Any faults detected here are reported (as described above) and MAP is locked.

MAP Lockup

If a MAP fault results from an execution unit request, MAP must take the following protective measures:

- Save the UTP.
- Block subsequent requests from being translated and creating multiple faults.
- Prevent the offending request from altering central memory.

The MAP saves the UTP by locking RMA and UTP registers by MAP-locked flip-flop. A 6-clock time-out allows time for the fault signal to reach execution unit trap mechanism. Setting MAP-locked FF causes the following:

- Blocks further requests until unlocked.
- Blocks strobing RMA/UTP/VAL rank.

- Destroys function currently being translated only if it is an execution function. RNI functions within MAP are held until MAP is unlocked.

The MAP-locked FF can be sensed by the microcode as a branch condition. It indicates that the UTP is useable. The UTP can be read via the MAP sense data path (described earlier in this section).

When the time-out ends, MAP input is enabled to accept new execution requests, and the first such request clears MAP Locked FF. Hence, the UTP must be read before this function is issued. The RNI unit remains quiet until reinitiated with a branch initialize or reset RNI function (see RNI section). While MAP is locked, MAP sense mux is forced to select the UTP for convenience.

Loading the Files

Only execution unit functions load the three MAP files (associative, real, and validity) shown on MLBD MAP 2.0. The order of loading must be as follows:

1. Load Associative File, the first function of a set of functions loading the file, does the following:
 - Performs normal MAP address hashing using four bits from BN exclusive ORed with four bits from SEG field.
 - Saves this address in a register for use by the two functions below.
 - Forces use of this hash address until load sequence terminates.
 - Loads process segment number (SEG) and BN into appropriate section of the associative file (44 bits). File selection (one-of-four) is done per least recently used (LRU) described later in this section.
 - Saves parity bits for the BN portion of associative data in a rank of FFs. This first function does not save parity bits in the files.
 - Saves a full bit according to the 64-bit mode of operation: monitor full (MF) or job full (JF) described later in this section.

CAUTION

Once a load associative file function has been issued, it must be followed, in the correct order, by the other two load functions before other MAP functions are permitted. MAP must be tested for all quiet before issuing the load associative files.

2. Load Real File, the second function of the set, does the following:
 - Loads the 32-bit input BN data into real file per LRU.
 - Saves parity bits for this data in another rank of FFs (as above). Note that segment portion of input data must contain zeros.

3. Load Validity File, the last function of the set, does the following:

- Loads the 32-bit input BN data into validity file per LRU.
- Writes parity bits into the file per LRU. Both parity bits saved above, as well as bits for the 32-bit validity data, are written.
- Releases the file address locked in during the load associative file.

NOTE

The segment portion of the input data must contain zeros. A load sequence updates the LRU code.

CENTRAL MEMORY FUNCTION RECODE

The functions presented to MAP hold the control information for both MAP and CM. MAP's logic extracts and recodes the CM control into the correct functions. These recoded functions are listed with each MAP function in table V-5-1 below.

MAP functions not requiring a CM request provide a dummy recode of 1111. This keeps the correct PMF function and provides additional error checking.

TABLE V-5-1. RING AND KEY CHANGES DURING BRANCHES

Function	Rings Changed	Rings Checked	Keys Changed	Keys Checked
Instr 48	None	Must be LE.R2 and GE.R1	Old G.OR.New G \geq PG New L \geq PL	For equality or master
Jump	None	None	None	None
Exch	New PRN \geq PRN	Must be LE.R2 and GE.R1	New G \geq PG New L \geq PL	None
Call	Lesser R2/PRN	None	Old G.OR New \geq PG New L \geq PL	For equality or master
Return	New PRN \geq PRN	Must be LE.R2 and GE.R1	New G \geq PG New L \geq PL	Between new P and SDE G must be = or SDE is master L must be =

P REGISTER KEY MODIFICATIONS

The keys associated with P register are modified as follows.

Call/Branch

For a call or branch, P register G key is compared with the G lock of the called or branched-to segment as shown in figure V-5-6.

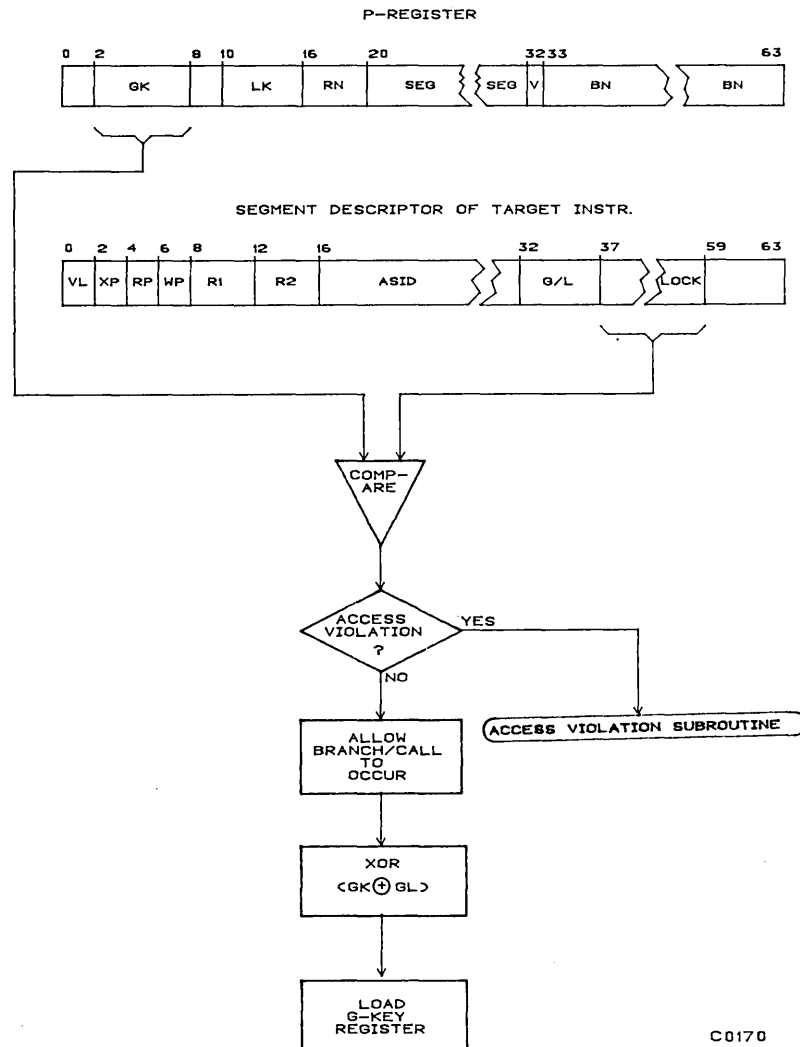


Figure V-5-6. Branch/Call Key Modifications

A call or branch is permitted if G key is equal to G lock, G master key, or G no lock. On a successful call/branch, P register is updated as shown in table V-5-2.

TABLE V-5-2. P REGISTER CALL BRANCH

Caller's G Key (Old P Register)	Callee's G Key/Lock (Segment Descriptor)	New G Key P Register
0	0	0
0	k2	k2
k1	0	k1
k1	k2	k1/k2*

* The new local L key of P register is always obtained from the callee's L key in the segment descriptor. Access violations result when k1 does not equal k2.

Return

On a return, P register local and global keys are obtained from the stack frame save area as shown in figure V-5-7.

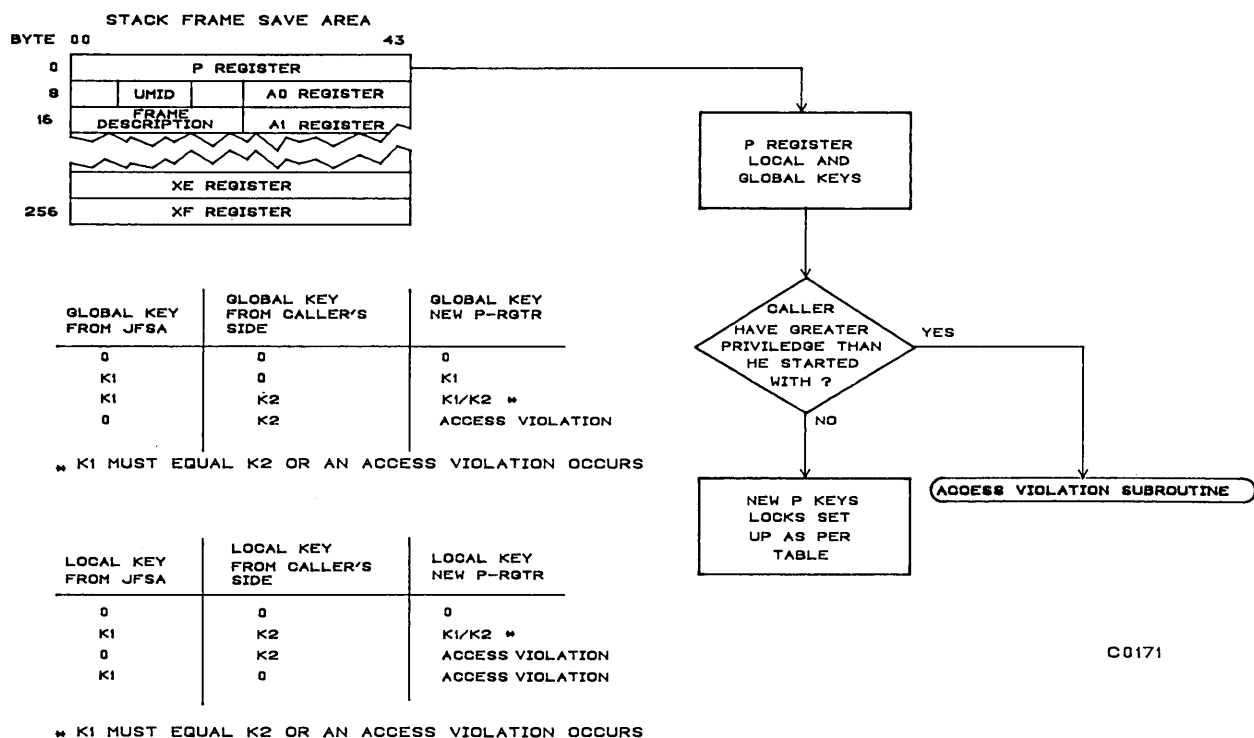


Figure V-5-7. Return Key Modifications

However, the hardware ensures that the caller does not return with greater privilege than s/he started with. See tables V-5-3 and V-5-4 for the setting of the new P register key/locks.

TABLE V-5-3. P REGISTER RETURN - G KEY

G Key from Stack Frame Save Area	L Key from Caller's Segment Descriptor	G Key New P Register
0	0	0
k1	0	k1
k1	k2	k1/k2*
0	k2	access violation
* k1 must equal k2 or an access violation results.		

TABLE V-5-4. P REGISTER RETURN - L KEY

G Key from Stack Frame Save Area	L Key from Caller's Segment Descriptor	L Key New P Register
0	0	0
k1	k2	k1/k2*
0	k2	access violation
k1	0	access violation
* k1 must equal k2 or an access violation results.		

The modification procedure is controlled by MAP function from execution as shown in the following paragraphs.

Jump (mnemonic IBRANCH JUMP, code = 3)

- Alters SEG and BN only
- Does not modify rings and keys
- Does not invoke ring/key tests

Branch (mnemonic IBRANCH INST, code = 4)

The old G key is compared to the G lock in the segment descriptor of the target instruction. An access violation fault is set if appropriate. If the

branch is permitted, the inclusive OR of the G key and the G lock forms and loads into the G key register. The L key is unconditionally loaded from the segment descriptor.

Call (mnemonic IBRANCH CALL, code = 5)

See explanation for Branch above.

Return (mnemonic IBRANCH RET, code = 6)

The G and L keys are loaded from MAP input data from execution unit and validity tested as above.

Exchange (mnemonic IBRANCH EXCH, code = 7)

The G and L keys are loaded from MAP input data with no validity checking.

P RING MODIFICATIONS

The P ring modification procedure (MAP 2.6) is controlled by MAP function from execution as shown in the following paragraphs.

Branch (mnemonic IBRANCH INST)

The P ring is not changed. The P ring is compared to the rings of the branch-to SDE (segment descriptor entry) and a fault is set if the P ring is less than R1 or greater than R2.

Call (mnemonic IBRANCH CALL)

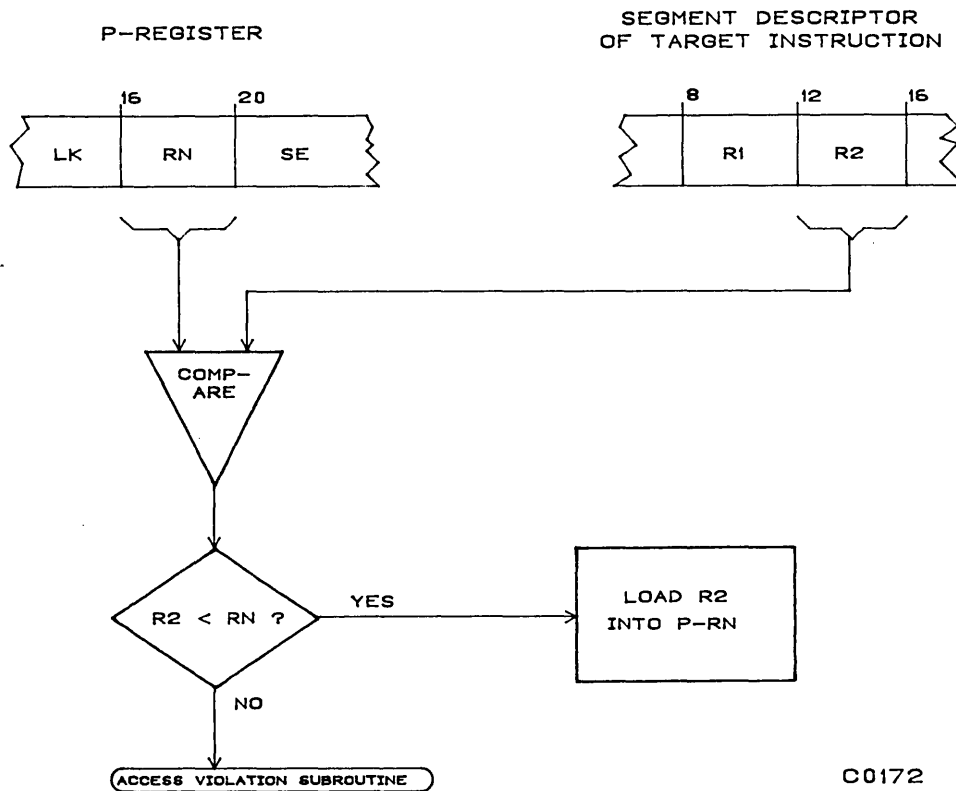
The current P ring is compared with R2 from the segment descriptor of the target instruction. If R2 is less than the current P ring, then R2 is loaded into the P ring as shown in figure V-5-8.

Return/Exchange (mnemonic IBRANCH RET/EXCH)

The P ring is unconditionally loaded from MAP input data from execution and is compared to the rings of the exchange-to SDE. A fault is set if the P ring is less than R1 or greater than R2.

MODIFIED BIT

In MAP the modified bit from page descriptor (MAP 2.1) enables all translation comparisons when MAP function specifies write validation. The M bit is loaded into the associative file at the same time as the PVA and occupies bit 1 of the input data. If an associative entry matches an address to be translated



C0172

Figure V-5-8. Call Ring Modifications

on a write validity request in all respects except for the M bit, the following actions occur:

- The hit is blocked; a MAP miss results.
- The matching entry is purged from MAP files.

A page table search results during which the microcode sets the M bit in the page descriptor in CM and sets bit 1 of associative file data during load-files sequence. Note that the modified bit is inverted when written into files.

LRU

A least recently used (LRU) field associated with each entry controls the writing of data into MAP files (MAP 2.1). Each file address (hash) has four entries. Each entry has a 2-bit LRU code associated with it. At each address, the LRU values must be 0, 1, 2, and 3 at all times.

Operation with All Files Enabled

Bits in the environment control register enables the operation of all MAP files as follows:

- The LRU codes at each file address are initialized to fixed values 0 through 3 by MAP function purge buffer per K, where K has values 8 through F.
- The four LRU codes at each file address are modified by a hit or load: LRU code of the entry is forced to zero; an LRU code greater than the old hit LRU is unchanged; an LRU code less than the old hit LRU increments by one.
- When loading the files, the data is written into the file which has an LRU equal to three. This LRU is then set to zero and the other LRUs are all incremented by one.

Operation With Files Disabled

Disabled files are described later in this section. When one or more files are disabled, MAP hardware calculates the following:

- Highest active LRU
- The entry with an LRU equal to the highest active

When initialized by a purge buffer function, the LRU for every set of entries is chosen so that the enabled files receive LRU values less than or equal to the highest active LRU value. Instead of the LRU equalling three, the file loading writes into the entry an LRU equal to the highest active LRU value.

MONITOR AND JOB ENTRIES

The use of separate full bits for 64-bit monitor and job mode entries is based on the following factors:

- The segment table address (STA) for the monitor process is established at the time of system deadstart and remains unchanged. Any segment translations (SEG to ASID) performed in monitor mode remain valid for the life of the system.
- If pages are deleted from the system page table, the software issues a purge buffer instruction to remove them from MAP files. Hence, any page translations (SVA to RMA) left in MAP remain valid until specifically purged by the operating system.
- Frequent exchanges occur between a job process and the monitor process followed by a return to the same job process. In this sequence, the segment table of the job process remains unchanged or expands: no entries are deleted. Hence, the segment translations previously performed for that job process are still valid.

To reduce exchange overhead caused by indiscriminate purging of MAP files, the monitor full and job full flags are used with each entry. An entry with the MF bit set is useable any time the processor is in monitor mode. An entry with the JF bit set is useable in job mode only if the segment table address of the process is the same as the STA of the process which placed the entry in MAP files. The following microcode procedures control MAP contents across process boundaries:

- On exchange from job mode to monitor mode, the files are left intact.
- On exchange from monitor mode to job mode, the STA of the new job is compared with the stored STA of the previous job mode process. If the same, the files are left intact; if different, a purge buffer per K (K = 0) MAP function is issued to purge the entries having the JF bit set. The new STA is then stored for subsequent testing on the next exchange from monitor to job.

MAP FUNCTIONS

The control information to MAP (MAP 2.3) is logically separated into two micrand fields: MAP request control (two bits) and MAP functions (five bits). These fields occupy bit positions 42 through 43 and 44 through 48 respectively of a Format D micrand. MAP functions are detailed in table V-5-5.

TABLE V-5-5. MAP FUNCTIONS (Sheet 1 of 4)

Mnemonic	CTL	Code	Description	CM FCN	Notes
-	0	ALL	NO-OP function ignored	-	
READ REQ NF	1	0	Read, no FLC check	0	
READ SEG	1	1	Segment table read	0	14
READ PAG	1	2	Page table read	0	14
READ PRE NF	1	4	Read pretest, no FLC check	-	15
READ RESYN	1	8	Read and resync CM	B	
READ INT	1	C	Read and send external interrupt	C	
WRITE REQ NF	1	10	Write, no FLC check	2	
WRITE PRE NF	1	14	Write pretest, no FLC check	-	15
READ MULT NF	1	18	Initialize STM, read, Xlate, no FLC check	0	9
WRITE MULT NF	1	1C	Initialize STM, write, Xlate, no FLC check	2	10
-	2	3	Load P register (alternate)	-	16
-	2	4	Load P register (alternate)	-	16
-	2	5	Load P register (alternate)	-	16
-	2	6	Load P register (alternate)	-	16
IBRANCH LOADP	2	7	Load P register	-	16

TABLE V-5-5. MAP FUNCTIONS (Sheet 2 of 4)

Mnemonic	CTL	Code	Description	CM FCN	Notes
READ PRE NM	2	11	Read pretest, no modulo check	-	15
WRITE PRE NM	2	12	Write pretest, no module check	-	15
READ PRE BS	2	14	Read pretest, binding section check, module 8 check	-	15
READ PRE M8	2	15	Read pretest, module 8 check	-	15
RWRITE PRM NM	2	18	Read and write pretest, no modulo check	-	15
WRITE RMA RANK	2	1	Write RMA data rank to clear errors	-	
IBRANCH LOAD P	2	7	Load P register, no translate	-	16
READ REQ RMA	3	0	RMA read, no Xlate	0	
WRITE REQ RMA	3	1	RMA write, no Xlate	2	
PURGE BUF	3	2	Purge buffer per K	-	1
IBRANCH JUMP	3	3	Jump branch	0	2
IBRANCH INSTR48	3	4	INSTR48 branch	0	2, 13
IBRANCH CALL	3	5	Call branch	0	2
IBRANCH RET	3	6	Return branch	0	2
IBRANCH EXCH	3	7	Exchange branch	0	2, 12
READ MULT RM	3	8	Initialize STM, read, RMA	0	8
LOAD SET	3	9	Load FLC and set 60-bit mode	-	
LOAD CLR	3	A	Load FLC, F latch and clear both	-	3
LOAD PSM	3	B	Load PSM (both copies)	-	4
LOAD ASS	3	C	Load associative file and hold address	-	5
LOAD REA	3	D	Load real file	-	5
LOAD VAL	3	E	Load validity file and clear address	-	5
LOAD F-L	3	F	Load and select F latch	-	
READ MULT NM	3	10	Initialize STM, read, xlate	0	9
READ REQ NM	3	11	Xlate, read validate	0	
WRITE REQ NM	3	12	Xlate, write validate	2	
-	3	13	Xlate, read, no validate	0	6
READ REQ BS	3	14	Xlate, binding section validation modulo 8 test	0	
READ REQ M8	3	15	Xlate, read validate, modulo 8 test	0	
WRITE REQ M8	3	16	Xlate, write validate, module 8 test	2	
WRITE MULT NM	3	17	Initialize STM, write, xlate	2	10
LOAD RNI	3	18	Reload RNI Register after stream (Reset)	-	
READ SET PV	3	19	Read and set lock, PVA	4	
READ CLR PV	3	1A	Read and clear lock, PVA	5	

TABLE V-5-5. MAP FUNCTIONS (Sheet 3 of 4)

Mnemonic	CTL	Code	Description	CM FCN	Notes
READ SET RM	3	1B	Read and set lock, RMA	4	11
READ CLR RM	3	1C	Read and clear lock, RMA	5	
READ EXC	3	1D	Read exchange address register	9	
READ FRC	3	1E	Read free running counter	A	
READ INT	3	1F	Initialize STM, WRITE, RMA	2	

NOTES:

- The value K is expected to be in bits 60 through 63 of the input data K controls the purge operation as follows:

K = 0 Purge job mode entries (described earlier).

K = 1-7 Undefined.

K = 8, 9, B-F Purge the entire MAP.

K = A Purge the four entries at the hash address defined by SEG and BN.

NOTE

A purge operation does not write data into the entire file. The written data includes the segment number and the M bit in associative file; the full bit is cleared. To write data into the rest of associative, validity, and real files, use MAP functions 3C, 3D, and 3E. During deadstart, all files must be written with these functions to provide correct parity. A purge operation then must be done to clear the full bits.

The state of BN bit 32 further controls the purge operation as follows: if this bit equals 1, SEG bits 28-31 are temporarily substituted for the environment control bits which enable MAP files; the MF and JF bits are not cleared. This allows specific LRU settings for microcode diagnostics.

- All branches initialize RNI unit as described in RNI section. The differences in MAP interpretation of the four branch instructions involve key-lock and ring checking, and manipulation as described earlier in this section.
- Function A loads the input data (BN portion) into both FLC and F latch. However, it disables the control signals so that both registers are disconnected from use.

TABLE V-5-5. MAP FUNCTIONS (Sheet 4 of 4)

4. The copies of the page size mask must be formatted as described earlier in this section.
5. Sequencing, other constraints, and data formats are described earlier in this section.
6. Intended for use by RNI unit only.
7. Subsequent code used by RNI stream equals one. (RMA write).
8. Subsequent code used by RNI stream equals zero. (RMA read).
9. Subsequent code used by RNI stream equals eleven. (PVA read).
10. Subsequent code used by RNI stream equals twelve. (PVA write).
11. Used to read the 60-bit mode address held in a CM register as a result of the last IOU exchange request (60-bit mode). Function 9 above is unique to M1.
12. The exchange function should also be used for 60-bit mode branches, executive state relative branches, and restarting instructions. The exchange function does not perform any key or ring modification.
13. Intended for use by intersegment branch, reference code 048.
14. Segment table and page table reads translate as normal RMA reads. However, to recording hit rate data they generate special signals to the hardware performance monitoring facility.
15. Pretest operations do not generate CM requests. Normal validity tests are performed and the LRU values are updated. The address in RMA rank at the end of a pretest function which had a hit in MAP files is the translated address.
16. Loads P register without causing a request to memory; no translation or validity checking occurs.

The ROMs are read twice for each MAP function: on first cycle and any subsequent cycles (figure V-5-9). The bit pattern read out during the first cycle is held in a register until used in the second or subsequent cycle.

ROM ONEROM TWO

BIT NO. =	0	1	2	3	4	5	6	7
FIRST CYCLE	*WRITE	REQ CMC	XLATE	READ VAL	WRITE VAL	BS VAL	MOD 8	INIT STM
*FIRST CYCLE	STROBE NEXT	RNI2	RNI5	RNI6	CSU0	CSU1	CSU2	CSU3

C0173

NOTES

1. The symbol * means NOT.
2. RNI 2, 5, and 6 are MAP function code bits for storage in the next function register.
3. CSU 0, 1, 2, and 3 are CM function code bits used with this request.
4. INIT STM causes stream unit initialization.
5. Bits *Write, Req CMC, and Xlate control the mode of MAP operation for this function.
6. Other bits control the type of validity checking done for this function.

Figure V-5-9. MAP Control ROMs Format

MAP SENSE

CAUTION

If a possibility exists that MAP is currently performing a translation for RNI or execution, do not use the function MISC MAP which forces data selection within MAP. A test for MAP all quiet must precede this function. See Branch Conditions later in this section.

Under microcode control a 64-bit data path (MAP 2.5) from MAP is used to copy data from within MAP to execution units AD register as shown in figure V-5-10. The miscellaneous microcode function MISC MAP activates the path. The functions AD.Sel MAP or BD.Sel MAP copy the data into AD or BD. The data selection within MAP uses bits 20 through 31 (SEG field) of MAP input data from execution unit. If the read map function is not present, P register is selected to MAP sense data path by default and may be copied into AD or BD at any time by the functions AD.Sel MAP or BD.Sel MAP. The use of select bits 20 through 31 is as follows:

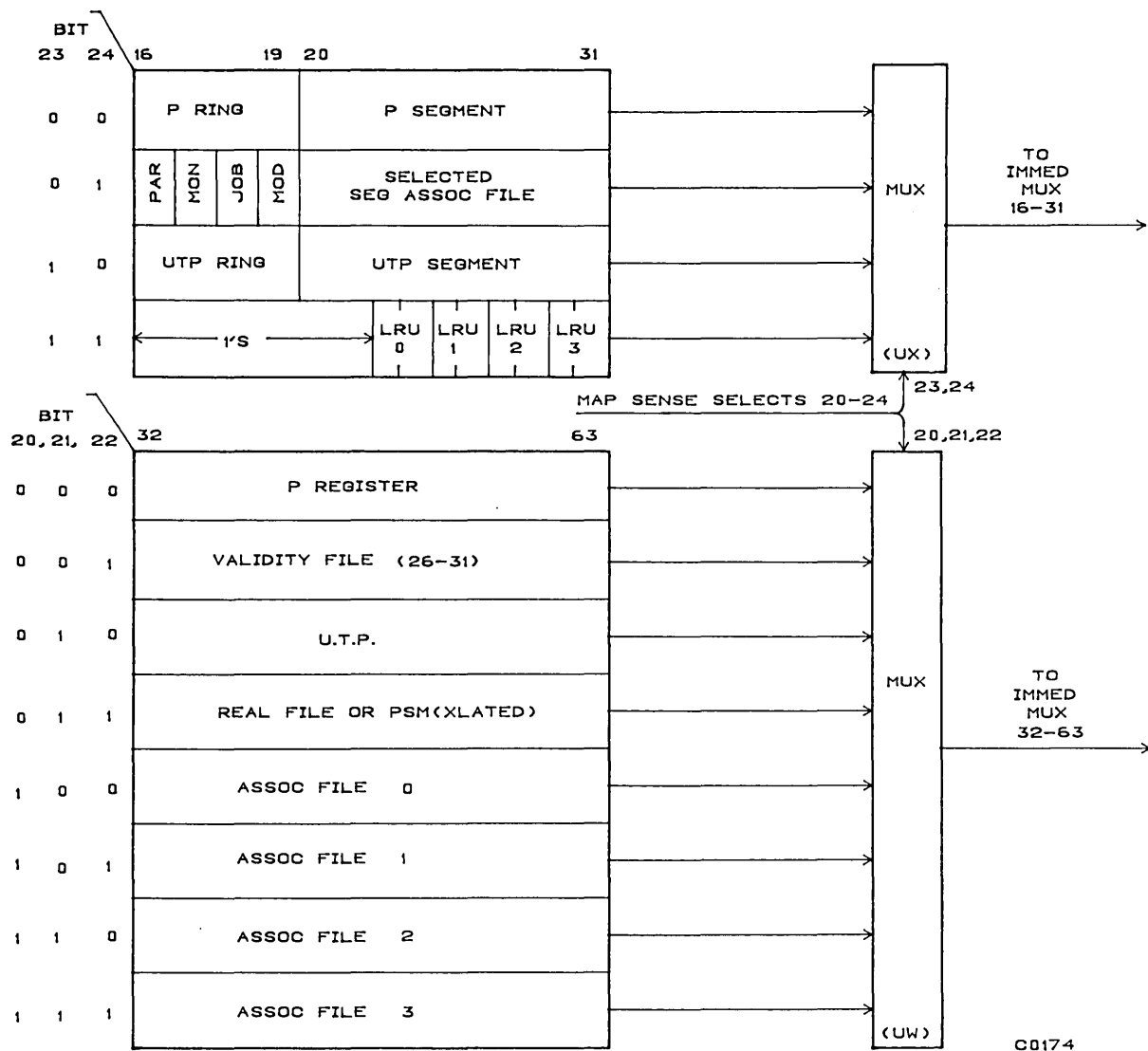


Figure V-5-10. MAP Sense

Bits	Selection
20-22	BN data selection (described later in this section)
23-24	RN, SEG data selection (described later in this section)
25	Unused
26-27	File number, if applicable (0-3)
28-31	File address, if applicable (0-15)

Key Field Sense (Bits 0-15)

This field of MAP sense data is not affected by the select bits described above. It always contains the current P register keys.

RN, SEG Field Sense (Bits 16-31)

The contents of this field are controlled by bits 23 and 24 of the select data. When these codes select data from the files, the file address and selection come from bits 26 through 31 defined above. The selections made by bits 23 and 24 are detailed in figure V-5-11. Note that the default is code 00.

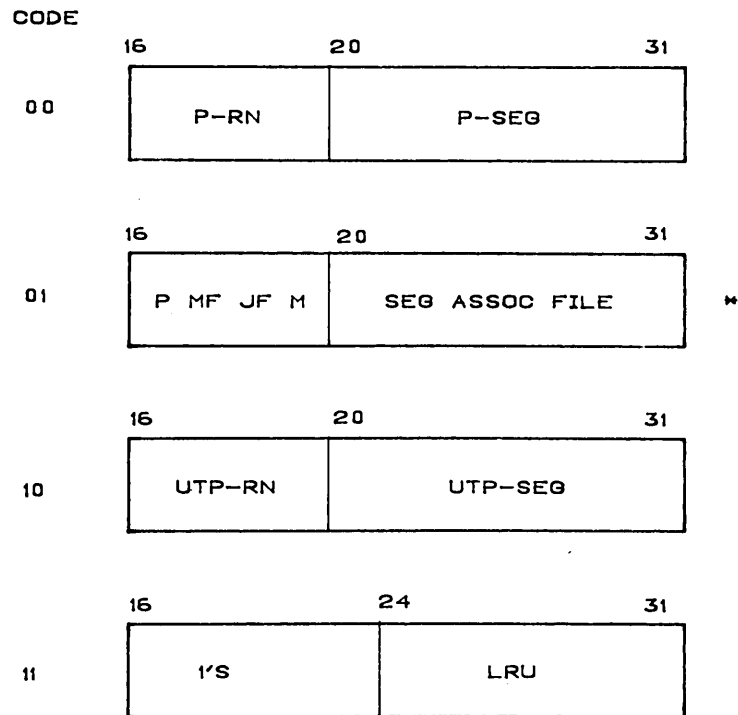
BN Field Sense (Bits 32-63)

Bits 20 to 22 of the select data control the contents of this field. When these codes select data from the files, the file address and selection come from bits 26 through 31 as defined above. The selections by bits 20 through 22 are detailed below. Note that the default is code 000.

<u>Code</u>	<u>Selection</u>
000	P register (BN portion)
001	Validity file, per bits 26-31
010	Untranslatable pointer (BN portion)
011	Translated address (see note below)
100	Associative file 0, address per bits 28-31
101	Associative file 1, address per bits 28-31
110	Associative file 2, address per bits 28-31
111	Associative file 3, address per bits 28-31

NOTE

Code 011 is a method for reading real file or the PSM register (BN copy). To read real file, set the input BN associated with the select codes to all 1's (or the PSM must already be set to all 1's). The translated address then becomes a direct copy of real file data. Conversely, if the real file data is set to all 1's, the translated address is a copy of BN or PSM.



C0175

NOTES

In code 01, the data items P, MF, JF, and M are defined as follows:

- P** The parity bit for the SEG portion of the associative file data. It is the exclusive OR of bits 20 through 31, MF, JF, and M. This bit may be used for fault isolation.
- MF** The monitor full bit indicates that the associated entry is full and contains a translation address for the monitor mode program.
- JF** The job full bit is similar to MF, but the translation address is for a job mode program (described earlier in this section).
- M** The modified bit from the page descriptor loads into the associative file from bit 16 of the input data during a load associative function.

Figure V-5-11. RN, SEG Field Sense Decode

BRANCH CONDITIONS

Branch conditions as shown in (table V-5-6) are available for the microcode to test internal states of MAP (illustrated on MLBD MAP 2.5).

TABLE V-5-6. MAP BRANCH CONDITIONS

Name	Mnemonic	Code	Description
Multiple hit	MULT	0	Multiple file hit condition
P contents invalid	BADP	1	P contents invalid
MAP locked	MAP	2	UTP register holds an address which cannot be translated
All quiet	ALL	3	No outstanding execution or RNI requests
BIG P	BIG	4	P register erroneously incremented by an instruction exit in the presence of a MAP fault
WRITE Miss	WRI	5	UTP currently locked up is from a write request
Global Privilege	GLO	6	Current segment of execution XP=11
Local Privilege	LOC	7	Current segment of execution XP=10 or 11

EXIT CONDITION

The MAP provides signals which control micrand exiting (table V-5-7).

60-BIT MODE FEATURES

A comparator to test addresses against FLC is the only feature provided in MAP to support 60-bit mode operation. Addresses presented to MAP in 60-bit mode must already be biased by RAC. The FLC comparator register must be loaded with the sum of RAC + FLC for the current job. Note that the P register in 60-bit mode contains a PVA which has been biased by RAC.

Loading the FLC register enables the checking of all subsequent PVA requests to MAP. Faults are reported as described earlier in this section.

TABLE V-5-7. MAP EXIT CONDITIONS

Name	Mnemonic	Description
MAP Empty	EXIT MAP	MAP input rank is empty and can accept a request.
Quiet MAP	EXIT QUI	MAP is not currently executing execution unit requests for either translation or validity testing.
CSU Data Ready	EXIT CSU	MAP decodes returning ident code from CM and produces this signal; indicates data is being returned to execution unit.
RMA Empty	EXIT CSU	Data rank associated with MAP's RMA rank is empty; can accept a data word.
<p>NOTE:</p> <p>The empty exit conditions indicate that the specified request can be honoured in a particular clock cycle. To use the empty exit conditions, the request (address or data) is sent in the same micrand as the exit condition. In the clock cycle when the appropriate resource becomes available, the honouring of the request and the exiting of the micrand occur in parallel.</p>		

MAINTENANCE FEATURES

The MAP design provides features to:

- Inspect contents of registers and files under microcode control (as described earlier in this section).
- Enable/disable any of the files. The translation files are organized with four independent entries in parallel. Though any two entries may be disabled, the normal product set microcode continues to produce correct results at a lowered performance level. Bits in environment control register disable files, enable hits for the appropriate files, and affect the organization of LRU codes on a purge buffers function (described earlier in this section).

The environment control bits are:

<u>Bit</u>	<u>File Enabled</u>
26	0
27	1
28	2
29	3

Refer to note on purge buffers function under MAP Functions in this section in which SEG 28 through 31 replaces these EC bits.

RMA MODE

Setting environment control register bit 25 forces the MAP into RMA mode. All MAP functions are operational in RMA mode but are modified as follows:

- The address presented to the CSU (and hence placed in the RMA rank, UWRM) is an exact copy of the input BN in UWBN, regardless of the function issued to MAP.
- The input ring is selected into the ring voting circuit, rather than the larger of R1.
- Ring R2 is not copied into P register on branch instructions.
- All validity testing is disabled.

The two MAP control paks operate as follows:

UX

- Modifies ring compare R1:RN to force copy of RN into compare register for execution. UX is strobed on every execution function.
- Modifies ring compare PRN:R2 to inhibit error on branch + return + exchange.
- Blocks setting of ident bit 3 (RNI fault).
- Blocks reporting of all fault testing.

GA

- Blocks Xlate Second Cycle to UX, which inhibits updating LRU and purging of entry from a file having a write MISS
- Forces selection of BN into RMA

- Blocks error signal Xlate, No Hit (normally causes lockup)
- Blocks testing of validity file parity

MAP KILL

The miscellaneous microcode function MAP Kill can place RNI and MAP in a defined state at any time. This function does the following:

- clears any request being translated by MAP
- clears all RNI functions (P0 and P1, requestors)
- unlocks MAP if locked

This function does not clear any requests in RMA rank or within CM.

APPENDIX A

GLOSSARY OF TERMS

GLOSSARY OF TERMS

A

=====

Refer to the American National Standards Institute (ANSI) Vocabulary for Information Processing Standard for terms not included in this glossary.

<u>Term</u>	<u>Definition</u>
A	ampere
A register	address register
ac	alternating current
AD	register file A data
ADS	register file A data mux select - (AD multiplexer select)
ADU	assembly/disassembly unit
ALN	arithmetic and logical network
ALU	arithmetic and logical unit
ANSI	American National Standards Institute
AOR	address out of range
ARVI mux	
ASCII	American Standard Code for Information Interchange
ASID	active segment identifier
ASX mux	register file A data sign extended mux (64-bit)
BAS	barrel and slot
B adder	big adder basic arithmetic
B mux	branch multiplexer
BC	base constant
BCD	binary coded decimal
BCOND	branch condition
BCR	branch on condition register
BD	branch delta
BDP	business data processing
BDS	BD multiplexer select
BFR	buffer
BKPT	breakpoint
BKPT register	breakpoint register
BN	byte number
BR	bounds register
BRA	branch
BS	binding section
CBP	code base pointer
CEJ/MEJ	central exchange jump/monitor exchange jump
CEL	corrected error log
CEM	configuration environment monitor
CF	critical frame
CFF	critical frame flag
CH	channel
CIF	CMU interrupted flag
CM	central memory
CMA	central memory access
CMC	central memory control
CMU	compare/move unit
COEFF, coeff	coefficient

CP	central processor
CPU	central processing unit
CRT	cathode ray tube
CS	control store
CSF	current stack frame
CSU	control store unit
CTI	common test and initialization
Clk	clock
Clr	clear
D1	decimal digit one
D2	decimal digit two
DBE	double bit error
dc	direct current
DCT	device control table
DEC	decimal; model-dependent environment control (see EC)
DI	debug index
DLP	debug list pointer
DM	debug mask
D mux	
DMR	debug mask register
DS	deadstart
DSC	display station controller
DSP	dynamic space pointer
DSR	data set ready
DTR	data terminal ready
EBCDIC	expanded binary-coded decimal interchange code
EC	environment control
ECC	error correcting code
ECL	emitter-coupled logic
ECM	extended central memory
ECS	extended core storage
ED	external device
EDS	extended deadstart
EI	environment interface
EIA	Electronic Industries Association
EID	element identifier
EM, EMS	exit mode selection field
EMC	exit mode condition field
EPF	external procedure flag
ER mux	ER multiplexer
ES	end suppression toggle (BDP edit instruction)
ESC	ESC execution sense field
EUAP	execution unit advance pipe
EXP, Exp	exponent
FF	flip-flop
FIFO	first-in, first-out
FL	field length
FLC	central memory field length register
FLE	extended core storage field length register
FP	floating point
FPE	floating point exception
FRC	free running counter
FS	fault status
FSM	fault status mask
FSR	fault status register

FS1	fault status 1
FUNC	floating point function field
FWA	first word address
Gen	generator
GE	greater than or equal to
GK	global key
GL	global lock
GT	greater than
HIVS	hardware initialization and verification
Hz	Hertz
I mux	instruction multiplexer
I/B	integer/Boolean
I/O	input/output
IC	integrated circuit
ICI	integrated controller interface
ILH	instruction look-ahead hardware
IOU	input/output unit
ISD	Supermini Disk Unit (Intelligent Small Disk)
ISEL	select code or I mux select bits
ISMT	Supermini Tape Unit (Intelligent Small Magnetic Tape)
JF	job full
JPS	job process state pointer
KC	keypoint code
KCN	keypoint class number
KEF	keypoint enable flag
KM	keypoint mask
L adder	large adder - double precision arithmetic (48-bit)
L mux	length multiplexer
LDB	load or store length determination box
LDS	long deadstart sequence
LE	less than or equal to
LED	light-emitting diode
LK	local key
LPID	last processor identification
LRN	largest ring number
LRU	least recently used
LSB	least significant bit
LSD	least significant digit
LSI	large scale integration
LSM	last symbol mask; load store multiple length
LT	less than
LWA	last word address
MA	monitor address field
MAC	maintenance access control
MAH	maintenance access hardware
MALADY	microcode assembler
MAP	modified addressing procedure
MAS	micrand address selection logic
MB	megabyte
MC	master clear
MCH	maintenance channel (unit)
MCI	maintenance channel interface
MCR	monitor condition register
MCU	maintenance control unit
MDF	model-dependent flags
MDHR	micrand diagnostic holding register, EMIT 64 register

MDW	model-dependent word
MF	monitor flag
MHR	micrand holding register
MLBD	multilevel block diagram
MMR	monitor mask register
MOP	micro-operator (BDP edit instruction)
MOS	metal-oxide semiconductor
MPS	monitor process state pointer
MR	maintenance register
MSB	most significant bit
MSD	most significant digit
MSNZB	most significant nonzero byte
mux	multiplexer, selector
NE	not equal
no-op	no-operation instruction
NORM	normalization
NOS	Network Operating System
ns	nanosecond
NS	negative sign toggle
N/U	not used
OCF	on-condition flag
OI	options installed
ON	occurrence number
OP or opcode	operation code
OS	operating system
OSB	operating system bounds register
Osc Sel	oscillator select
pak	printed circuit assembly package
P register	program address register
PCB	power control box
PE	parity error
PFA	page frame address
PFF	primitive full
PFS	processor fault status
PID	processor identifier
PIO	parallel input/output
PIT	process interval timer
PMF	performance monitoring flag
PMF	performance monitoring facility
PN	page number
PNR	point of no return
PO	page offset
PP	peripheral processor
PPM	peripheral processor memory
PPU	peripheral processor unit
PSA	previous save area
PSF	previous stack frame
PSM	page size mask e.g. PSM register
PTA	page table address
PTE	page table entry
PTL	page table length
PTM	processor test mode
PVA	process virtual address
PW	partial write
QTY	quantity

R adder	page offset adder
R mux	R multiplexer
R register	relocation register or return register
R1 register	return register 1
R2 register	return register 2
R/W	read/write
RAC	central memory reference address register
RAE	extended core storage reference address register
RAM	random access (read/write) memory
RF	register file
RFA	register file A
RFB	register file B
RFPW	register file partial write (field)
RI	radial interface
RMA	real memory address
RN	ring number
RNI	read next instruction
ROM	read-only memory
RP	read permission (segment descriptor field)
RTC	real time clock
RTS	request to send
S	streaming unit
S adder	small adder floating point arithmetic
S mux	control store address multiplexer
S register	control store address register
SAD	S adder sense condition
SBD mux	register file B data mux (64-bit)
SBE	single bit error
SCS mux	shift count select mux
SCT	special characters table (BDP edit instruction)
SDE	segment descriptor table entries
SDT	segment descriptor table
SECODED	single error correction/double error detection
SEG	process segment number
SEL	select
SFDR	stack frame descriptor register
SFSA	stack frame save area
SHC	shifter control (field)
SIT	system interval timer
SM	the symbol (BDP edit instruction)
SN	negative sign (BDP edit instruction)
SPCP	system power control panel
SPID	segment/page identifier
SPT	system page table
SRT	subscript range table
SS	status summary
STA	segment table address
STL	segment table length
SV	specification value
SVA	system virtual address
SW	switch
TCD	type code
TED	trap-enable delay
TEF	trap-enable flip-flop
TM	test mode

TOS	top of stack
TPM	two port multiplexer
TTL	transistor-transistor logic
UART	universal asynchronous receiver/transmitter
ubit	micrand bit
UCR	user condition register
UEL	uncorrected error log
UEL1	uncorrected error log 1
UEL2	uncorrected error log 2
UEM	unified extended memory
UM	user mask
UMR	user mask register
us	microsecond
USR	user status register
UTP	untranslatable pointer
UVMID	untranslatable virtual machine identifier
V	volt
VC	search control code (page descriptor field)
VL	segment validation (segment descriptor field)
VMCL	virtual machine capability list
VMID	virtual machine identifier
WD	write data
WDMX or WD mux	write data mux
WDR	word disassembly register
WMX	write mux select
WP	write access control (segment descriptor field)
XOR	exclusive OR
XP	execute access control (segment descriptor field)
XPKG	exchange package
ZF	zero field toggle (BDP edit instruction)
ZIF	zero insertion force

APPENDIX B

IOU INSTRUCTION SUMMARY

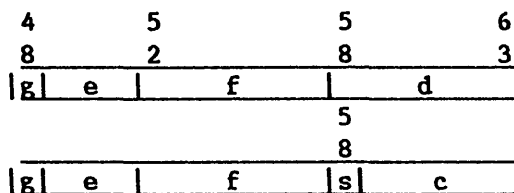
INSTRUCTION SET

The PP instruction set is based on the CYBER 170 PP instruction set. The 7-bit opcode includes the 6-bit opcode of the CYBER 170 PP. Extensions to the instruction set allow programs to manipulate 16-bit PP words and 64-bit CM words and to reference 28-bit CM addresses.

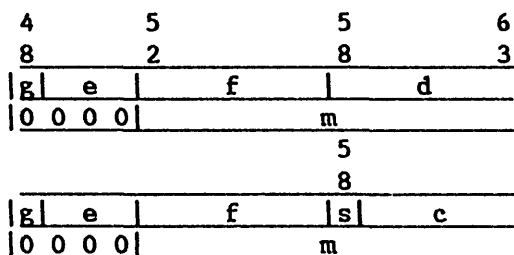
INSTRUCTION FORMATS

All PP instructions are represented in one of four formats. Two of these use a single 16-bit word; two use two consecutive 16-bit words. These formats are represented below.

16-bit formats



32-bit formats



The following field descriptions apply to both instruction formats:

- | | |
|-----------|--|
| f 6 bits | Least significant six bits of the 7-bit opcode. |
| d 6 bits | Operand, part of an operand, or address specification depending on the instruction. |
| c 5 bits | Channel number. |
| m 12 bits | Part of an operand, address specification, or I/O function code depending on the instruction. |
| g 1 bit | Most significant bit of the 7-bit opcode. In many instructions, g controls the width of the operand read from PPM. If g is 0, the operand is 12 bits; if g is 1, the operand is 16 bits. |

s 1 bit Subopcode used with certain I/O instructions.
0 Unused bits. Should be set to zero (reserved for future use).

The CYBER 180 PP instruction set in these formats includes the CYBER 170 PP instructions as a subset and therefore allows the execution of CYBER 170 PP programs without change in binary format.

12-BIT VERSUS 16-BIT EXCEPTIONS

The CYBER 170 PP instruction 27 x (read program address) operates as a pass instruction on the CYBER 180 IOU.

The CYBER 170 24 d and 25 d pass instructions execute as passes on the CYBER 180 IOU when the d-field is zero. Otherwise, they load and store the R register.

The CYBER 170 64 s c m, 65 s c m, 66 s c m, and 67 s c m instructions execute in the same manner on the CYBER 180 IOU when the s-field is clear. When s is set, the 64 and 65 instructions test and set or clear the channel flag bit and the 66 and 67 instructions test and clear the channel error flag bit.

ADDRESS MODES

Instruction operands are determined by the address mode of the instruction. Operands are available either from the instruction or from memory locations specified by the instruction. The operands may be 5, 6, 12, 16, or 18 bits long.

No-Address Mode

The no address mode uses the d-field directly as a 6-bit operand.

Constant Mode

The constant mode uses the d-field and the m-field directly as an 18-bit operand. The d-field contains the most significant 6 bits of the operand; the m-field contains the least significant 12 bits of the operand.

Direct Mode

The direct mode uses the d-field as the 6-bit address of a 12-bit or 16-bit operand in PPM.

Indirect Mode

The indirect mode uses the d-field as the 6-bit address of a word in PPM that is used as the address of a 12-bit or 16-bit operand in memory. Thus, d specifies the operand indirectly.

Memory Mode

The memory mode uses the d-field and m-field to specify the address of a 12-bit or 16-bit operand in PPM.

If the d-field is 0, bits 52 through 63 of the m-field are used as the address.

If the d-field is not 0, the d-field is the 6-bit address of a 12-bit index. This index is added to bits 52 through 63 of the m-field to generate the 12-bit address of all possible PPM locations (0 to 7777*).

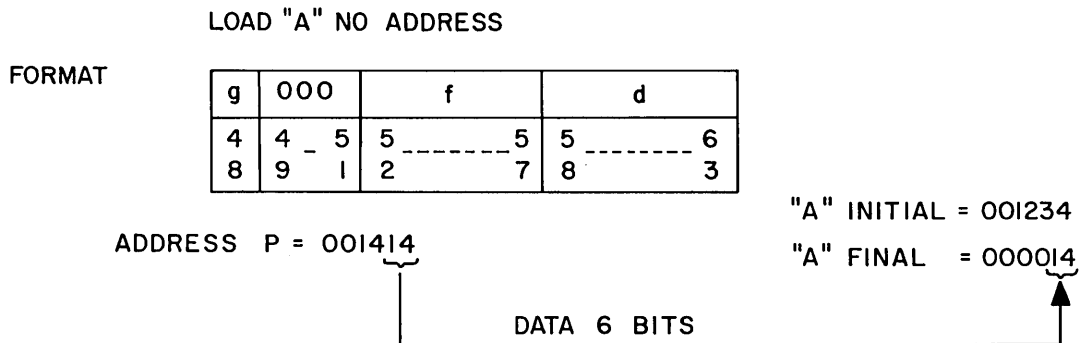
ADDRESS MODE INSTRUCTIONS

Figures B-1 through B-7 illustrate the addressing for the various instruction modes. Circled numbers in the figures show the instruction sequence.

Nomenclature used in the following instruction description is defined below:

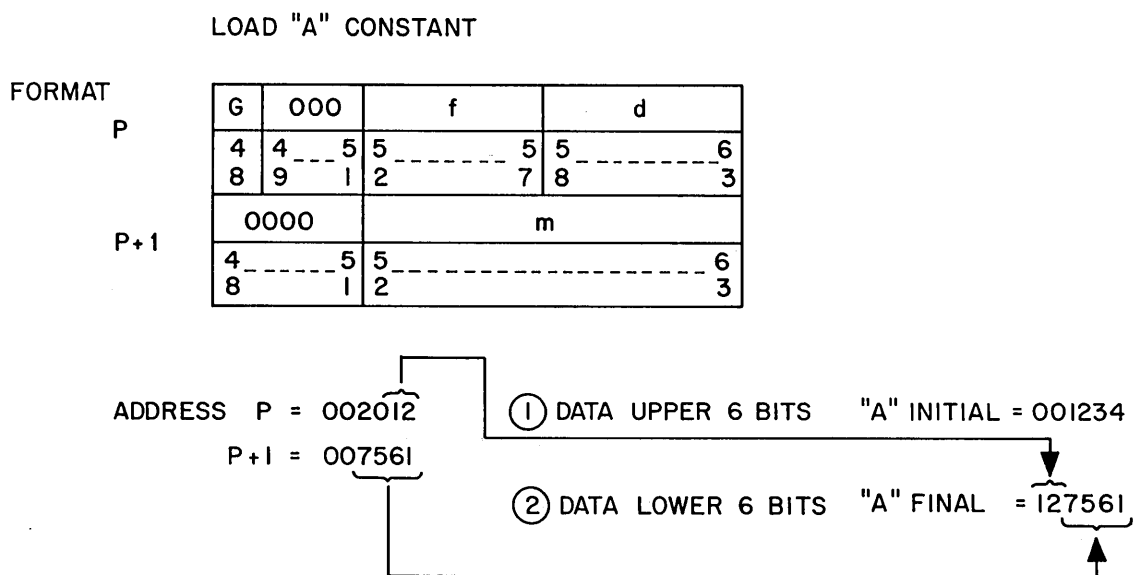
- A A register (Address register).
- (A) Contents of the A register.
- X One-bit field which can be either 0 or 1.
- c Channel number.
- d Value of the d-field (no address mode).
- dm The 18-bit value obtained from the d-field and the m-field (constant mode).
- (d) Contents of the location specified by the d-field (direct mode), except that (0)=0 if d=0.
- ((d)) Contents of the location specified by the contents of the location specified by d-field (indirect mode).
- m+(d) Address specified by the m-field indexed by the contents of the location specified by the d-field, except if d=0.
- (m+(d)) Contents of the location specified by the m-field indexed by the contents of the location specified by the d-field (memory mode), except if d=0.
- P P register (program address register).
- R R register (relocation register).
- (R) Contents of the R register.
- (R)+(A) CM address formed from the sum of the contents of the R and A registers.

* All PPM locations are in octal.



C0268

Figure B-1. No Address Mode Example



C0269

Figure B-2. Constant Mode Example

DIRECT INSTRUCTION EXAMPLE
12 BIT MODE

LOAD "A" DIRECT

FORMAT

	g	000	f	d
P	4	4 5	5 5	5 6
	8	9 1	2 7	8 3

ADDRESS P = 003020

① ADDRESS

PP
MEMORY
LOC =
17 = XXXXXX
20 = 123456
21 = YYYYYY
22 = ZZZZZZ

"A" INITIAL = 012345

"A" FINAL = 003456

② DATA 12 BITS

16 BIT MODE

ADDRESS P = 103020

① ADDRESS

PP
MEMORY
LOC =
17 = XXXXXX
20 = 123456
21 = YYYYYY
22 = ZZZZZZ

"A" INITIAL = 012345

"A" FINAL = 123456

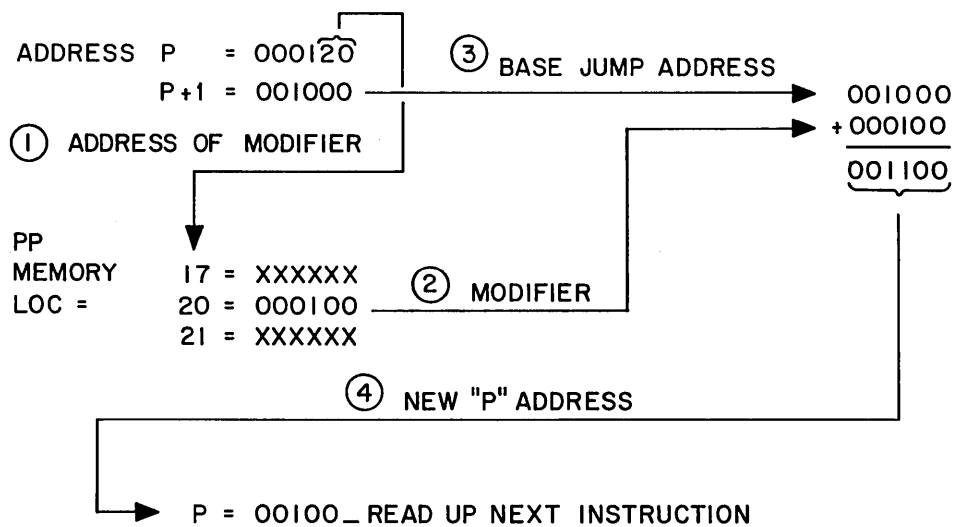
② DATA 16 BITS

C0270

Figure B-3. Direct Mode Example, 12- and 16-bit Instructions

FORMAT

P	g	000	f				d			
	4	4	5	5	5	5	5	6	6	6
	8	9	1	2	7	8	3	3	3	3
P+1	m									
	0000		5	2	6	3				



NOTE : IF d IS EQUAL TO ZERO THEN m (THE BASE ADDRESS) WOULD BE USED AS THE JUMP ADDRESS.

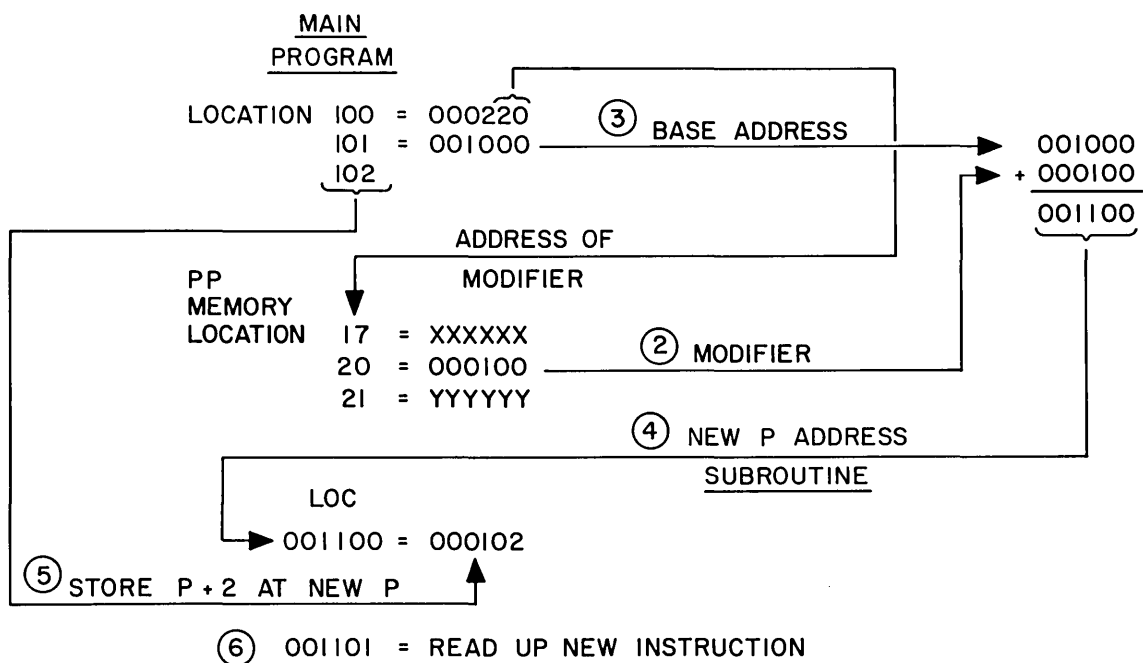
CO27I

Figure B-4. Direct Mode Example, Long Jump Instruction

RETURN JUMP EXAMPLE

FORMAT

P	g	000	f	d
	4	4 5	5 5	5 6
	8	9 1	2 7	8 3
P+1		0000	m	
	4	5	5	6
	8	1	2	3



NOTE : IF d IS EQUAL TO ZERO THEN m IS USED AS THE NEW P ADDRESS.

TO GET BACK TO MAIN PROGRAM YOU MUST JUMP TO THE ADDRESS AT LOCATION 001100.

C0272

Figure B-5. Direct Mode Example, Return Jump Instruction

INDIRECT MODE INSTRUCTION EXAMPLE

12 BIT MODE

LOAD "A" INDIRECT

FORMAT

P

g	000	f	d
4	4 _ 5	5 ----- 5	5 ----- 6
8	9 _ 1	2 ----- 7	8 ----- 3

LOC P = 004020

ADDRESS
OF

ADDRESS

PP
MEMORY
LOC =

17 = XXXXXX
20 = 001000
21 = YYYYYY
22 = ZZZZZZ

② ADDRESS

777 = XXXXXX
1000 = 123456
1001 = YYYYYY
1002 = ZZZZZZ

A INITIAL = 012345

A FINAL = 003456

③ DATA 12 BITS

NOTE : IF BIT 48 WERE A ONE THEN ALL 16 BITS WOULD BE
LOADED INTO "A".

IE: INST. = 104020 A FINAL = 123456

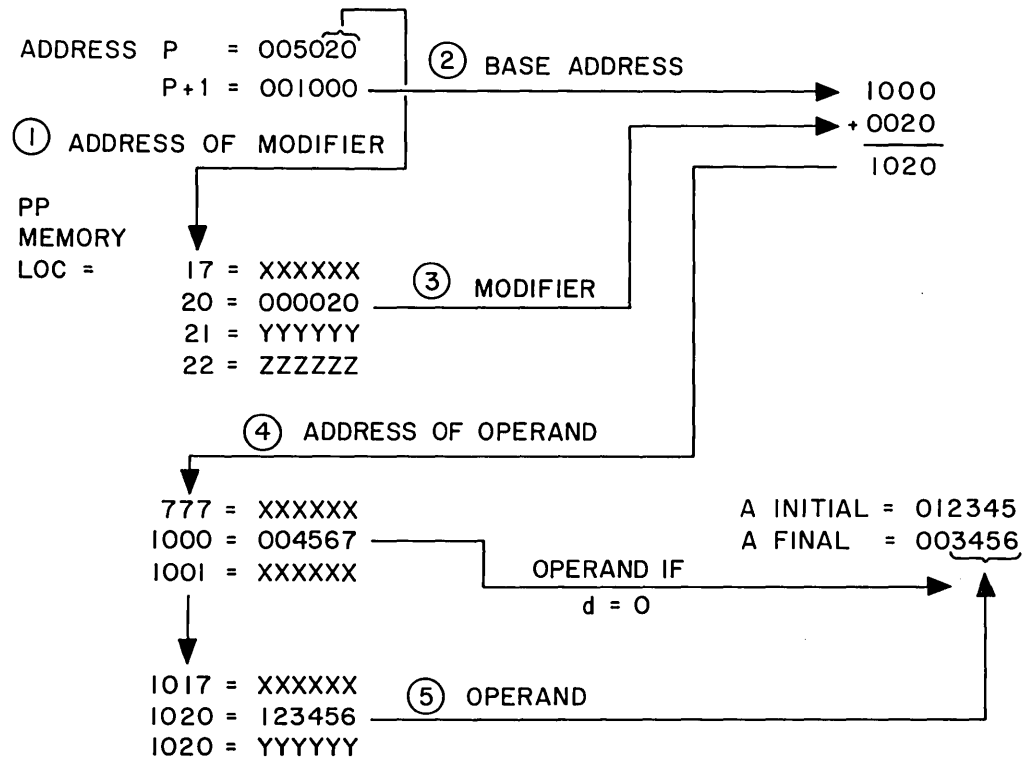
C0273

Figure B-6. Indirect Mode Example

MEMORY MODE INSTRUCTION EXAMPLE
LOAD 'A' MEMORY MODE

FORMAT

P	g	000	f	d
	4	4 5	5 5	5 6
	8	9 1	2 7	8 3
P+1	0000		m	
	4	5	5	6
	8	1	2	3



NOTE : IF d PORTION OF INSTRUCTION IS ZERO THEN THE BASE ADDRESS WOULD BE USED AS OPERAND ADDRESS AND 'A' WOULD BE LOADED FROM LOC. 1000.

IF BIT 48 IS EQUAL TO A ONE THEN ALL 16 BITS WOULD BE LOADED IN 'A'

C0274

Figure B-7. Memory Mode Example

SHORT WORD (12-BIT) STORES

The instructions which store the least significant 12 bits of a PPM word (0002, 0034 through 0037, 0044 through 0047, and 0054 through 0057) and the CM reads (0060 and 0061) clear the most significant four bits of the PPM words.

USE OF LOCATION 0

The four CM block transfer instructions (061, 063, 161, and 163) and the four block input/output instructions (071, 073, 171, and 173) make use of memory location 0 to save the value of the P register during instruction execution. The P register and associated incrementing logic function as the PPM address for the block transfers. The actual value stored in location 0 is the address of the m-field of the instruction, for example, P+1. When the instruction exits, the contents of location 0 is incremented by 1 and placed in the P register before the next instruction is executed, allowing normal instruction sequencing to resume.

If the contents of location 0 are altered during the instruction execution, the next instruction is executed out of the initial sequence at (0)+1. This action results from the block transfer of data from CM or a channel into location 0.

USE OF LOCATION 7777

Memory address mode instructions (m+(d))

In this mode, m plus (d) forms the address of the operand. This generates a 12-bit address within PPM. The following table indicates the addresses formed with the specified values of d and m.

	<u>m=0</u>	<u>m=7777</u>	<u>0<m<7777</u>
d=0	00	00	m
d NE 0, (d)=0	00	00	M
d NE 0, (d)=7777	00	7777	M
d NE 0, 0<(d)<7777	(d)	(d)	(d)+m

The first word address (FWA) for block I/O and CM access instructions is in the category 0 less than m less than 7777 in the example above.

UNUSED BITS

When one or more bits from an instruction are unused, that is, their value(s) and associated function(s) are not specified within the instruction description, the execution of these instructions is not affected by the values of the unused bits.

IOU FLOWCHARTS

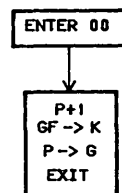
Figures B-8 through B-38 provide flowcharts for the PP instructions. Refer to the microcode listings in appendix C for details and accuracies. The following notes explain the conversions and terminology used.

NOTES FOR FIGURES B-8 THROUGH B-38

1. Each box in the flowcharts represents one trip of an instruction.
2. When there is the possibility of ambiguity, the sequence in which operations occur within a given trip is established by the order (top to bottom) in which they are written.
3. Refer to appendix C for A and Q adder codes.
4. The following symbols are defined as:

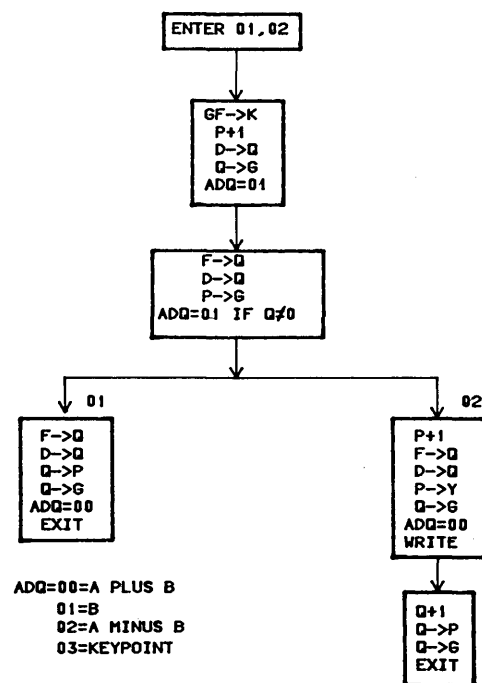
g,e,f,d	fields of PPM
Y	data in to PPM
G	address to PPM
Z	zero

5. $Q2^5$ equals Q58 on CE 3.0.



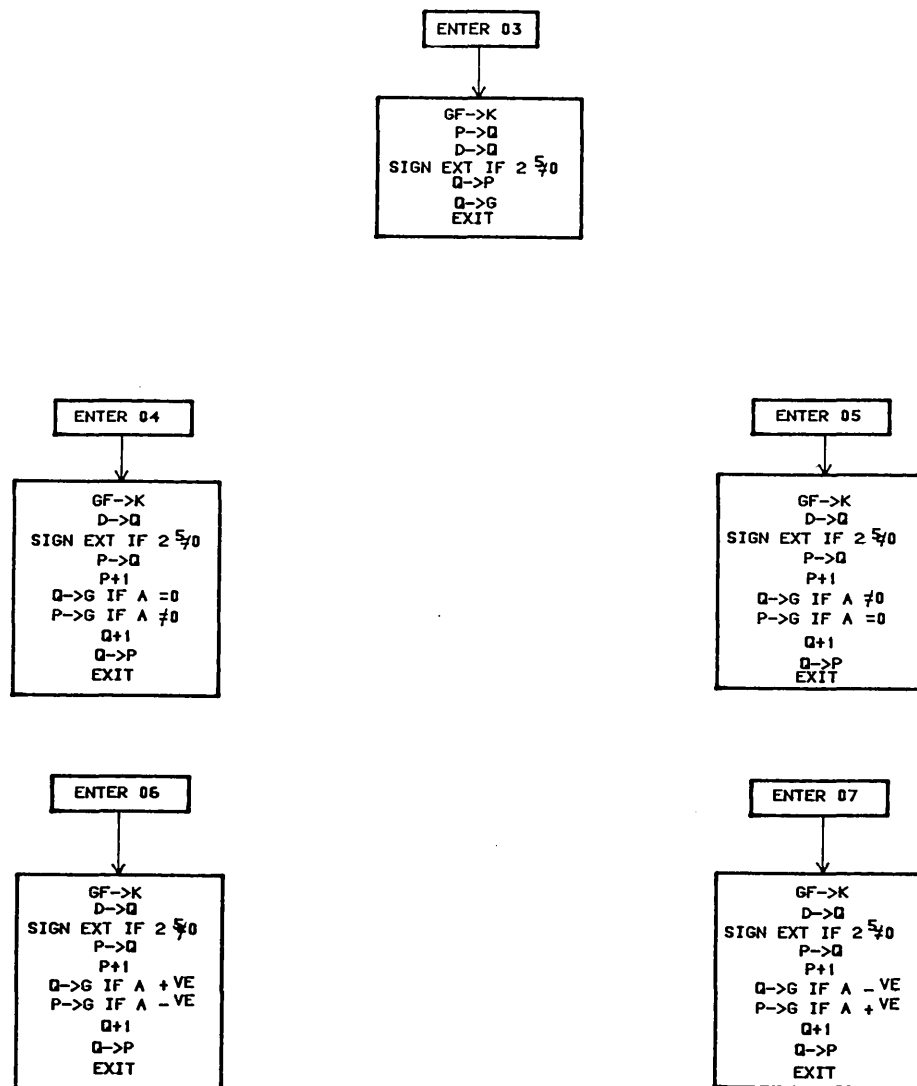
C1107

Figure B-8. Flowchart, PP Instruction 00



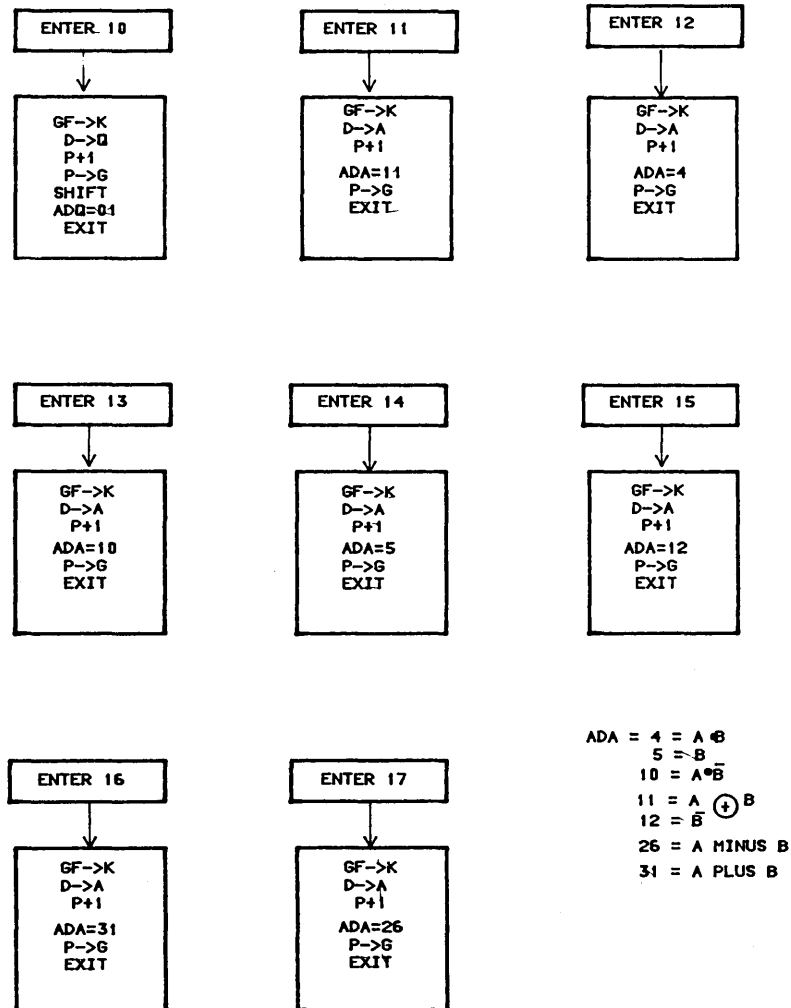
C1108

Figure B-9. Flowchart, PP Instructions 01 and 02



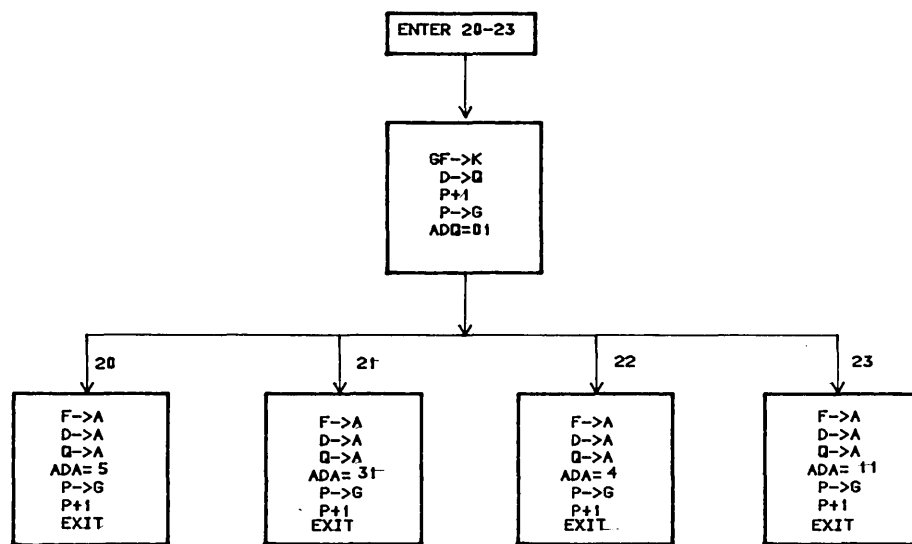
C1109

Figure B-10. Flowchart, PP Instructions 03 through 07



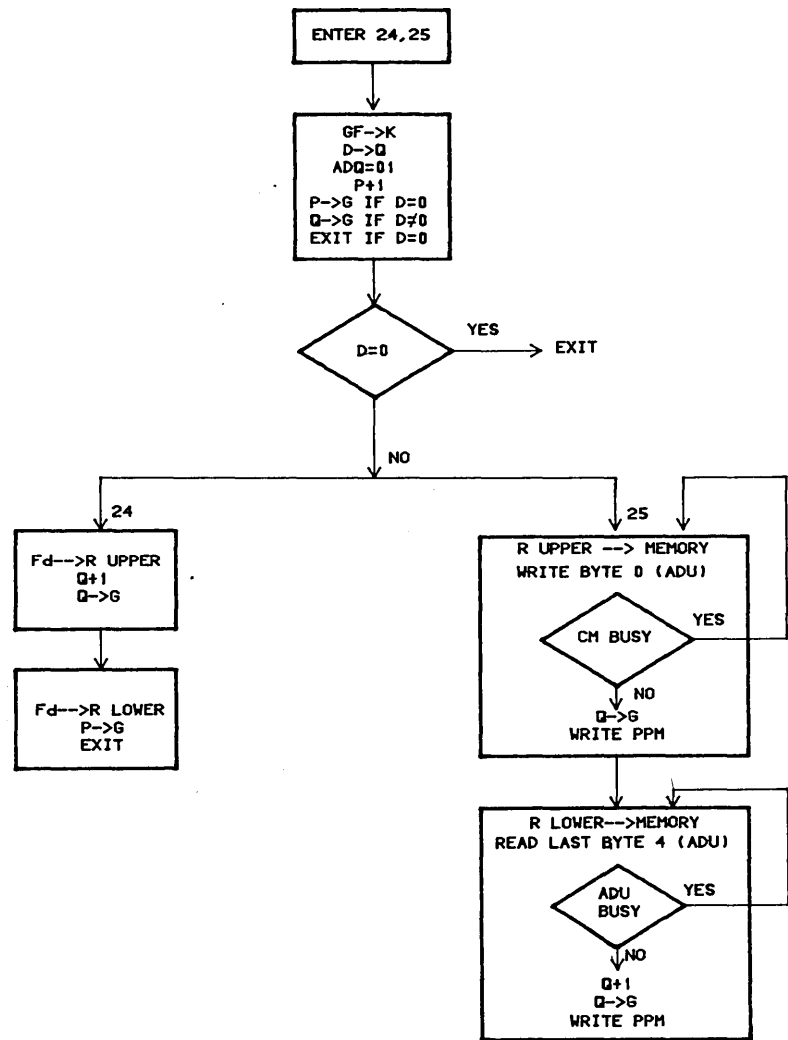
C1110

Figure B-11. Flowchart, PP Instructions 10 through 17



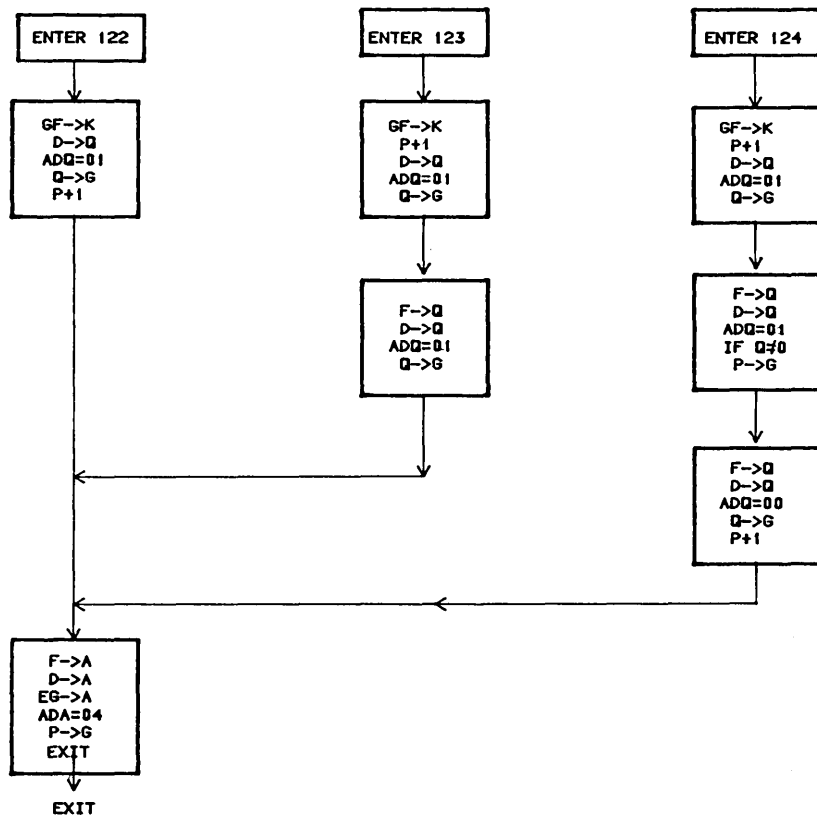
C1111

Figure B-12. Flowchart, PP Instructions 20 through 23



C1112

Figure B-13. Flowchart, PP Instructions 24 and 25



C1113

Figure B-14. Flowchart, PP Instructions 122 through 124

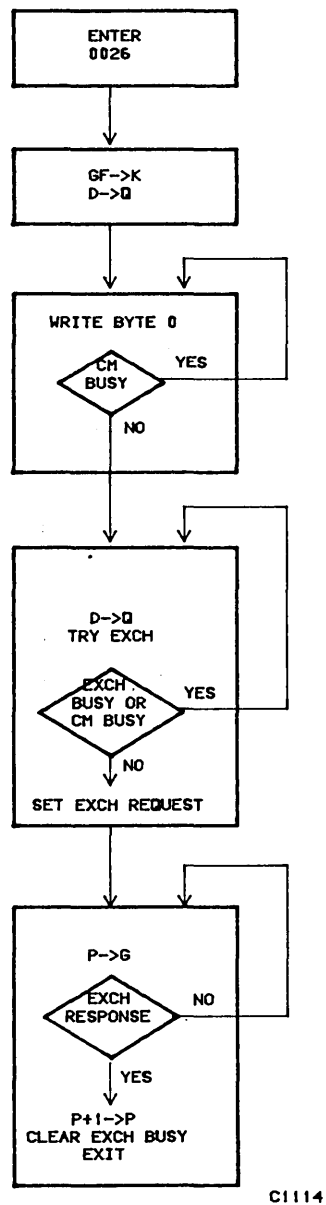
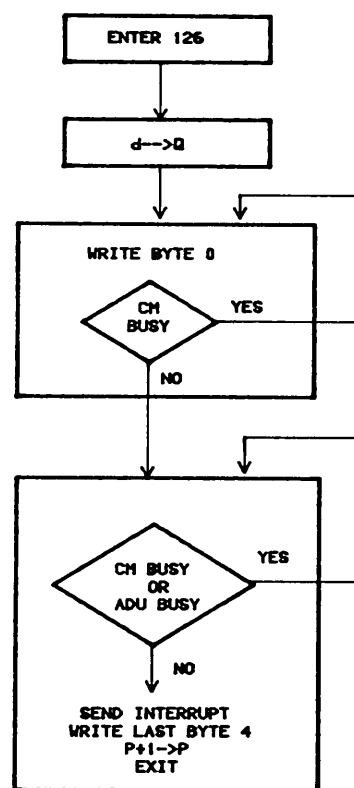
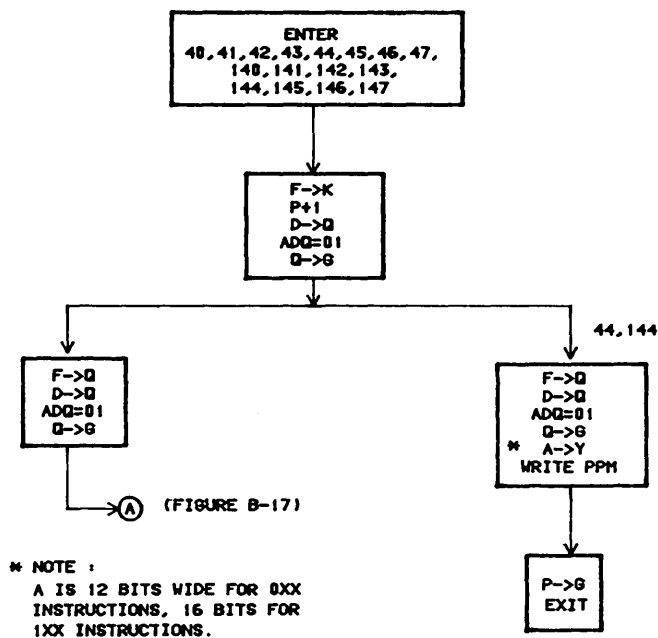


Figure B-15. Flowchart, PP Instruction 26



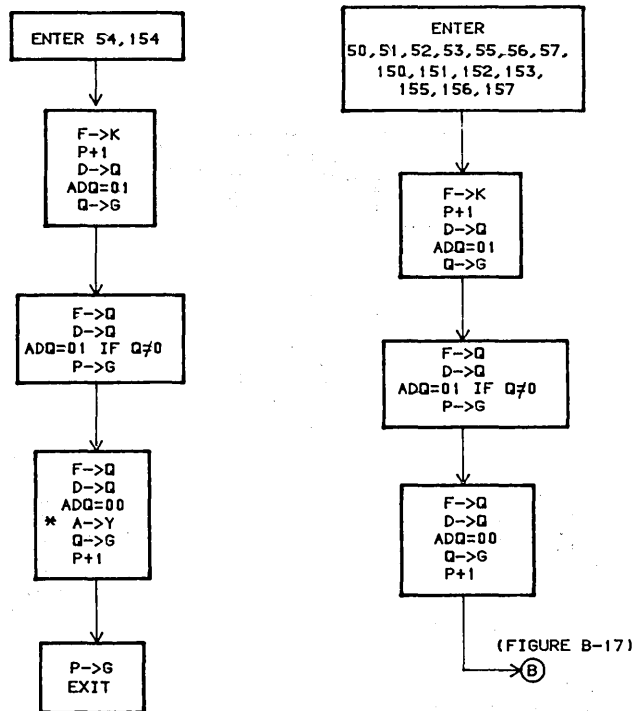
C1115

Figure B-16. Flowchart, PP Instruction 126



C1117

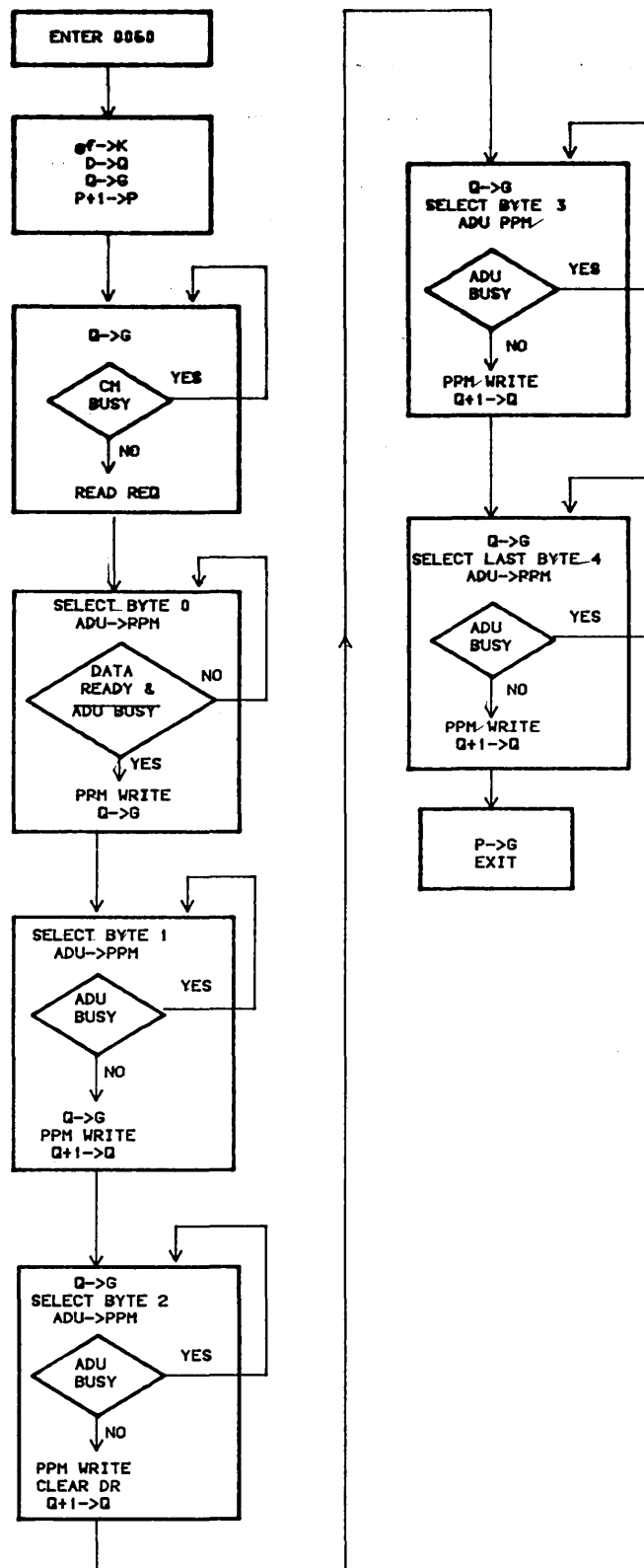
Figure B-18. Flowchart, PP Instructions 40 through 47, 140 through 147



* NOTE: A IS 12 BITS WIDE FOR 0XX INSTRUCTIONS, 16 BITS FOR 1XX INSTRUCTIONS.

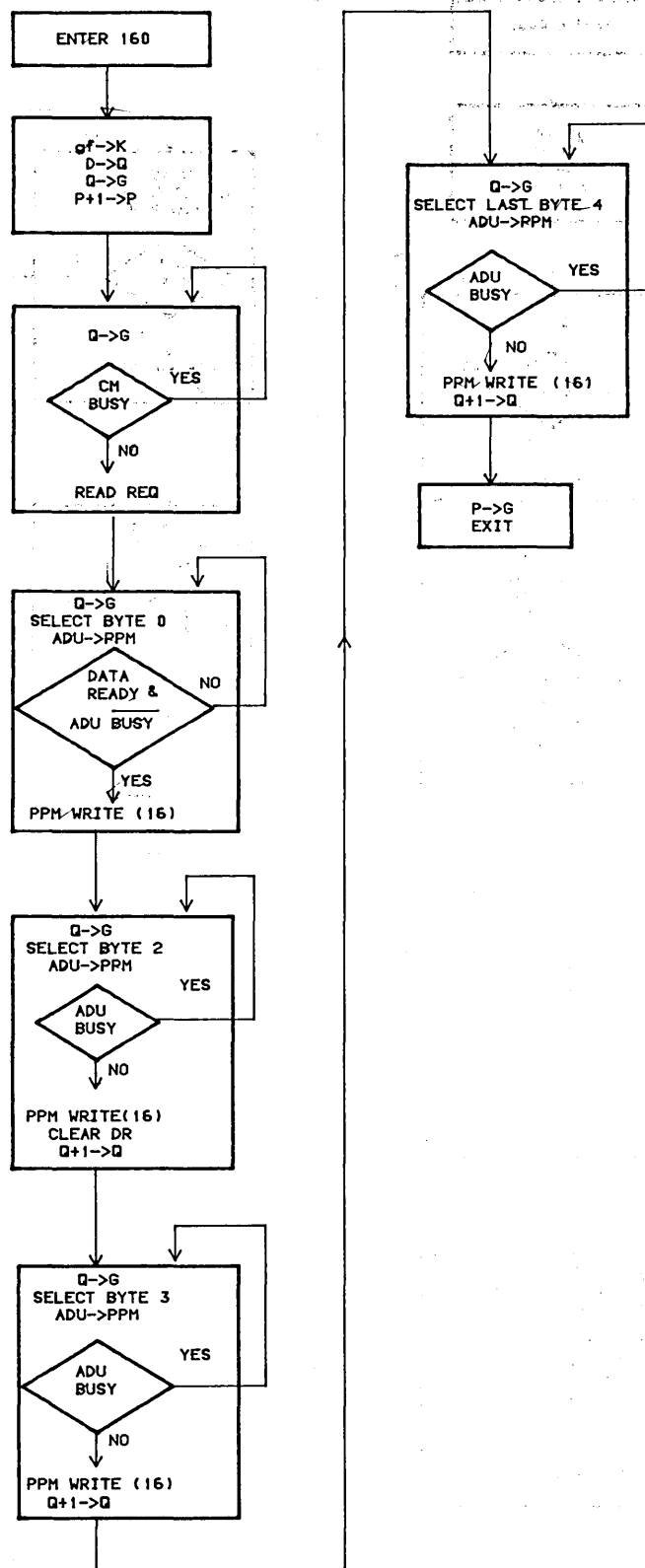
C1118

Figure B-19. Flowchart, PP Instructions 50 through 57, 150 through 157



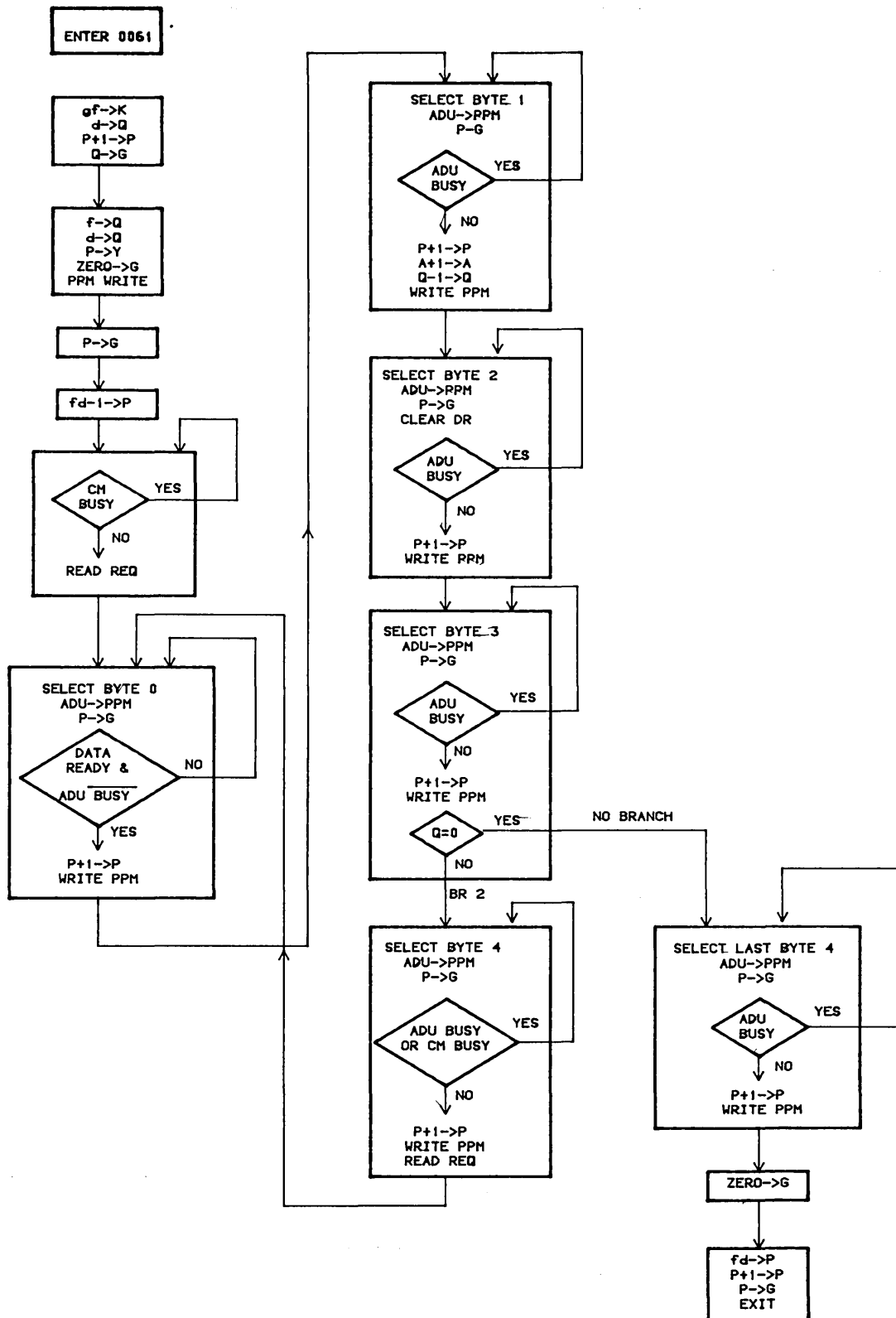
C1119

Figure B-20. Flowchart, PP Instruction 60



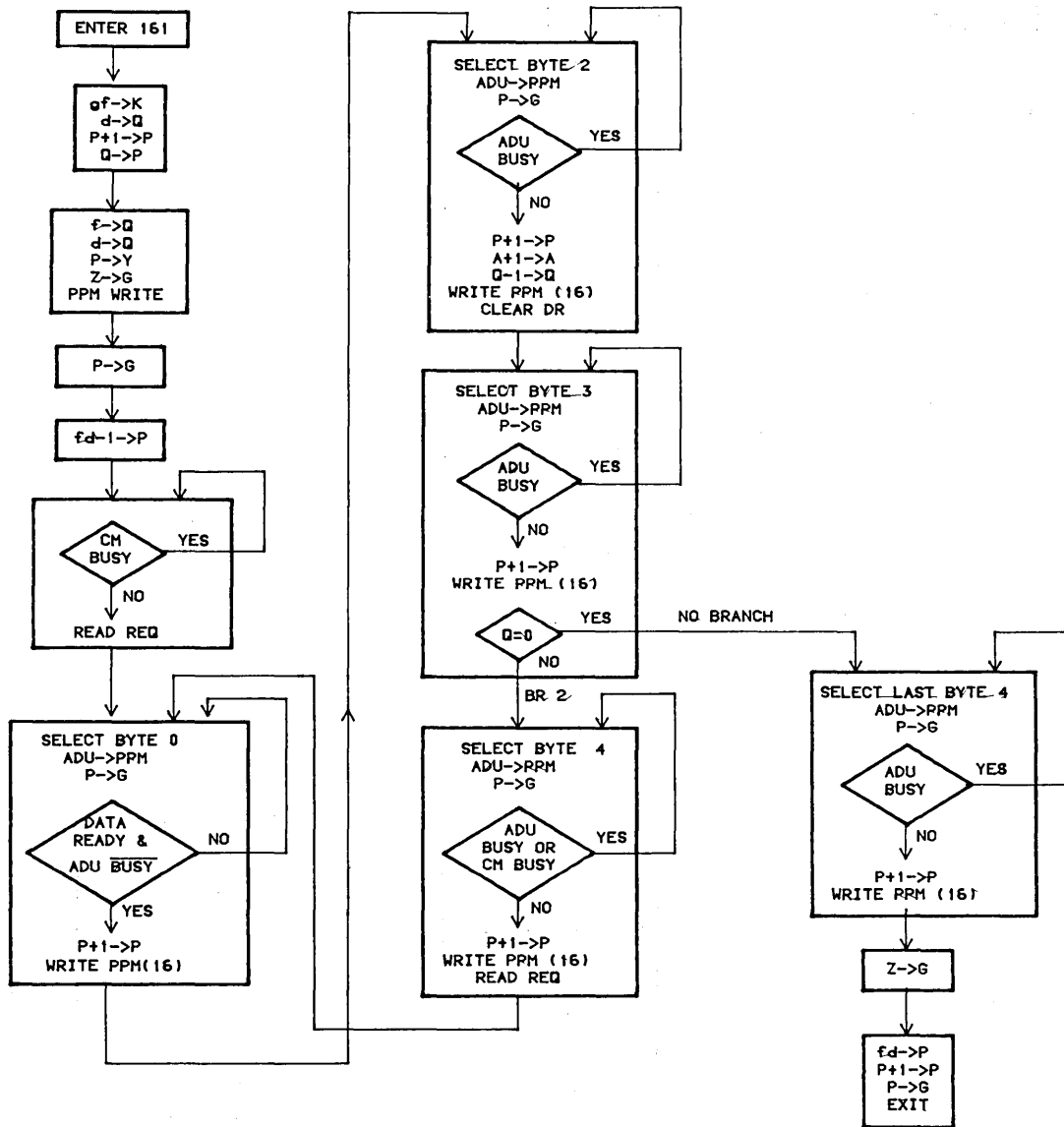
C1120

Figure B-21. Flowchart, PP Instruction 160



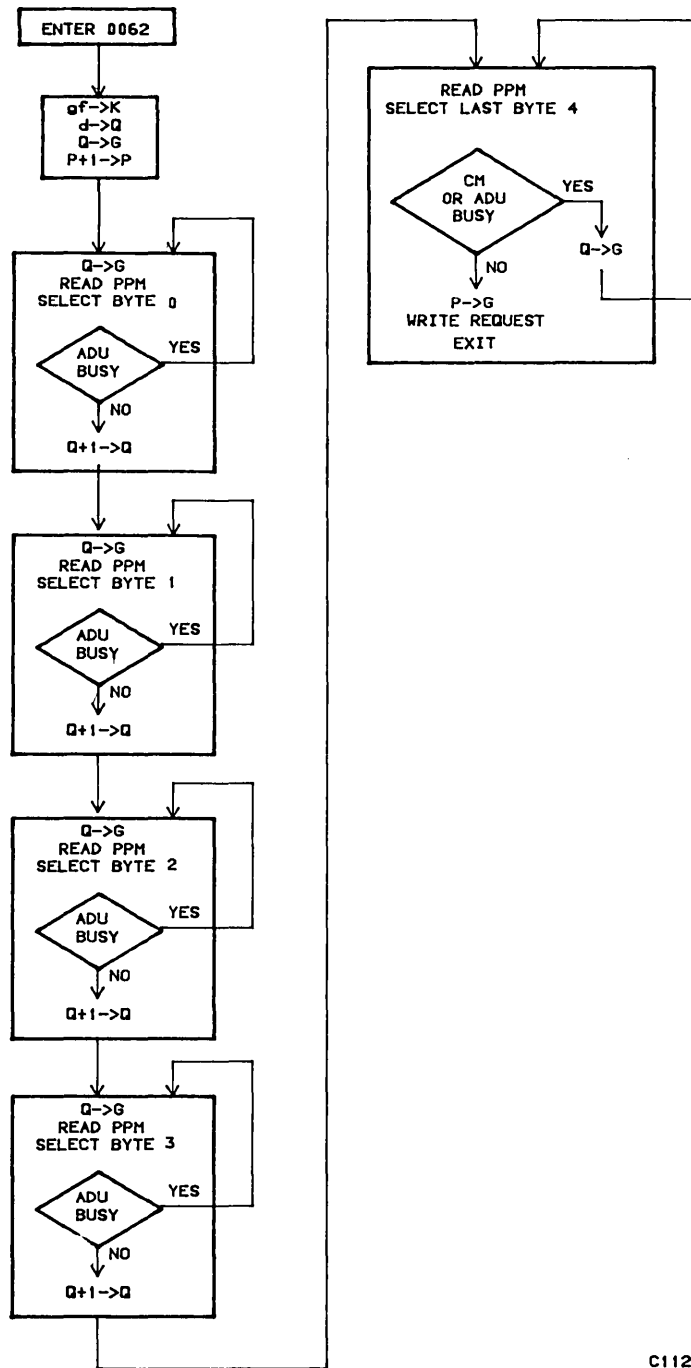
C1121

Figure B-22. Flowchart, PP Instruction 61



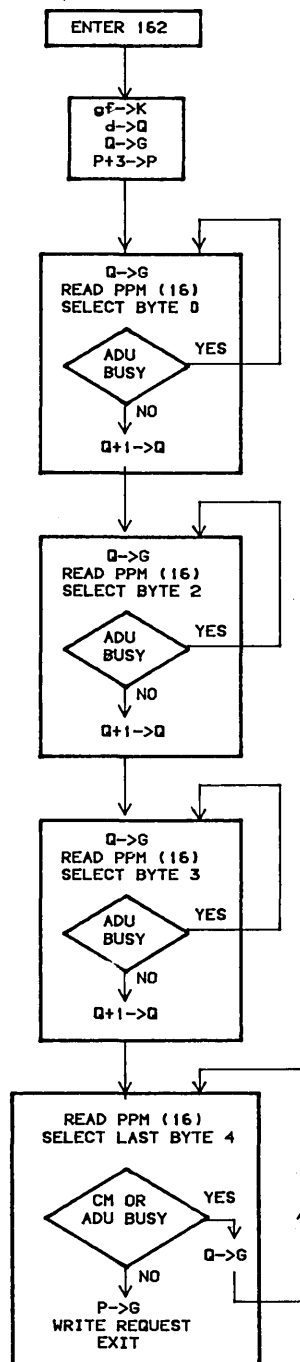
C1122

Figure B-23. Flowchart, PP Instruction 161



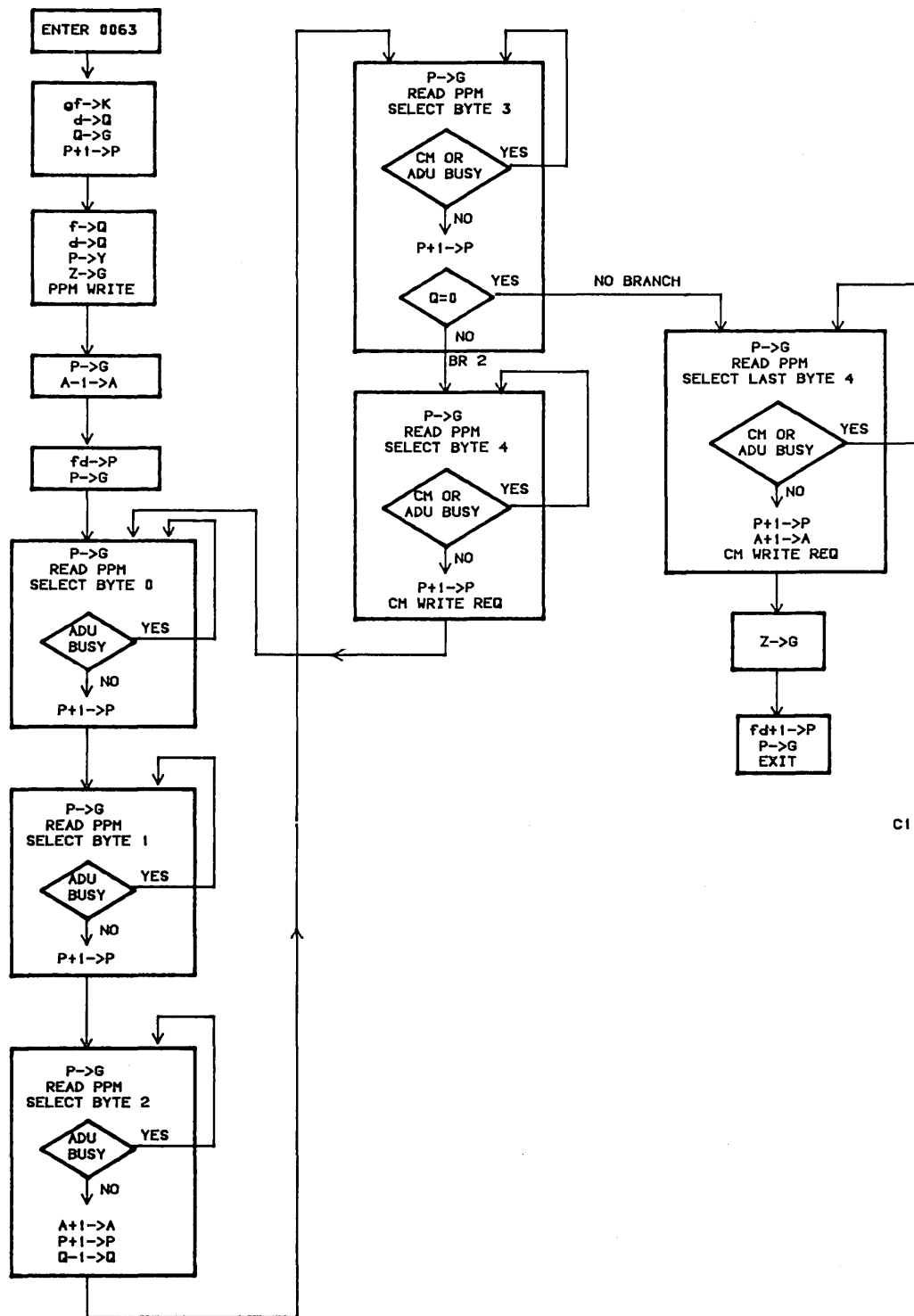
C1123

Figure B-24. Flowchart, PP Instruction 62



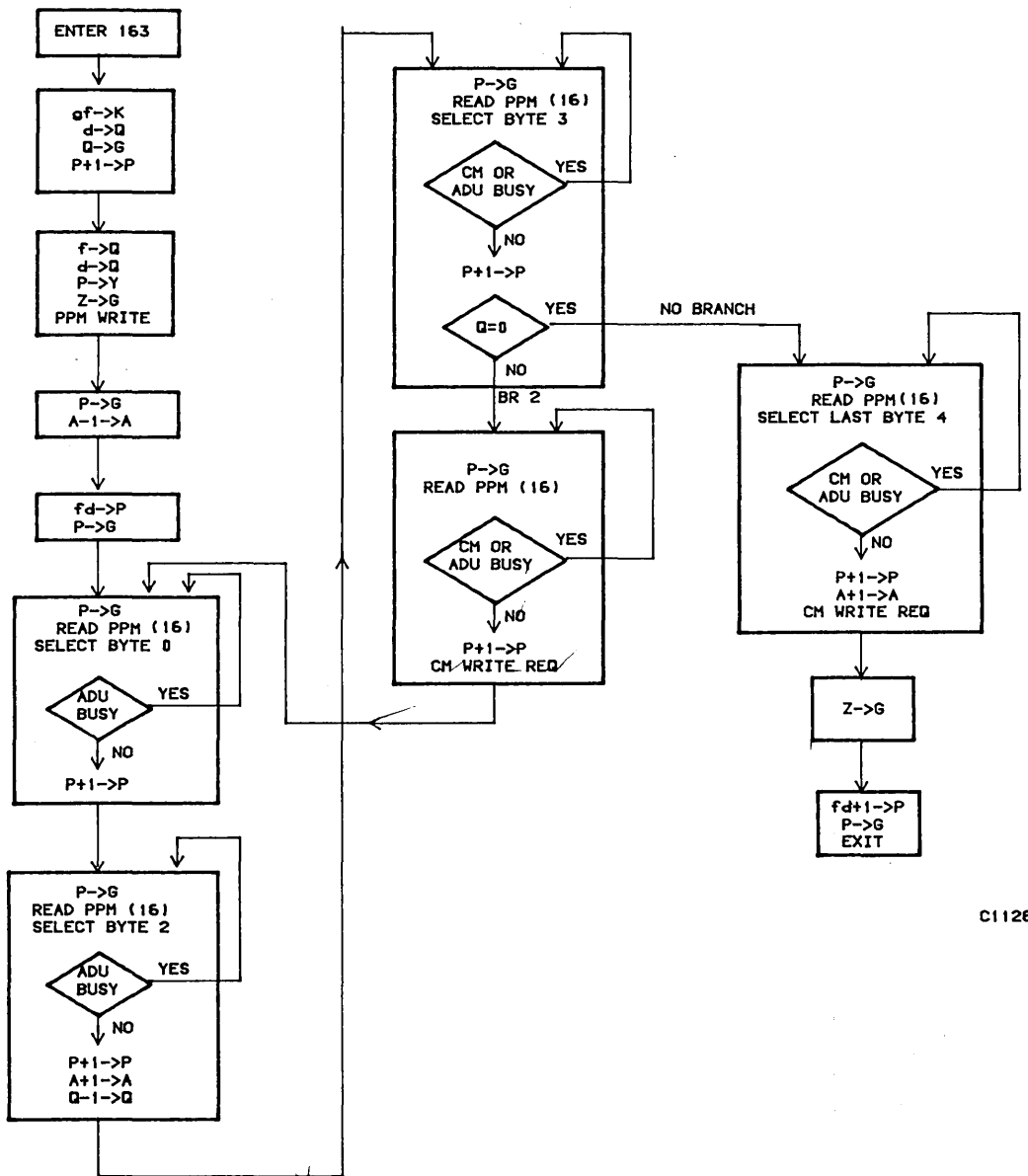
C1124

Figure B-25. Flowchart, PP Instruction 162



C1125

Figure B-26. Flowchart, PP Instruction 63



C1126

Figure B-27. Flowchart, PP Instruction 163

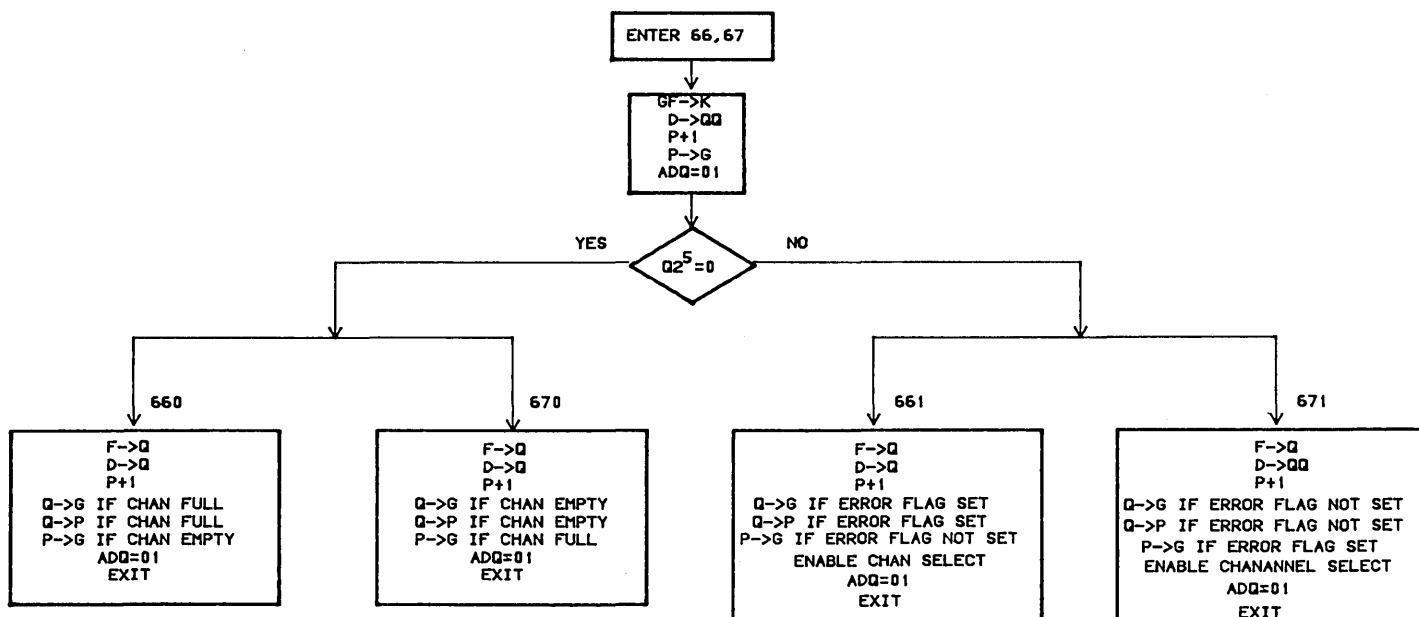
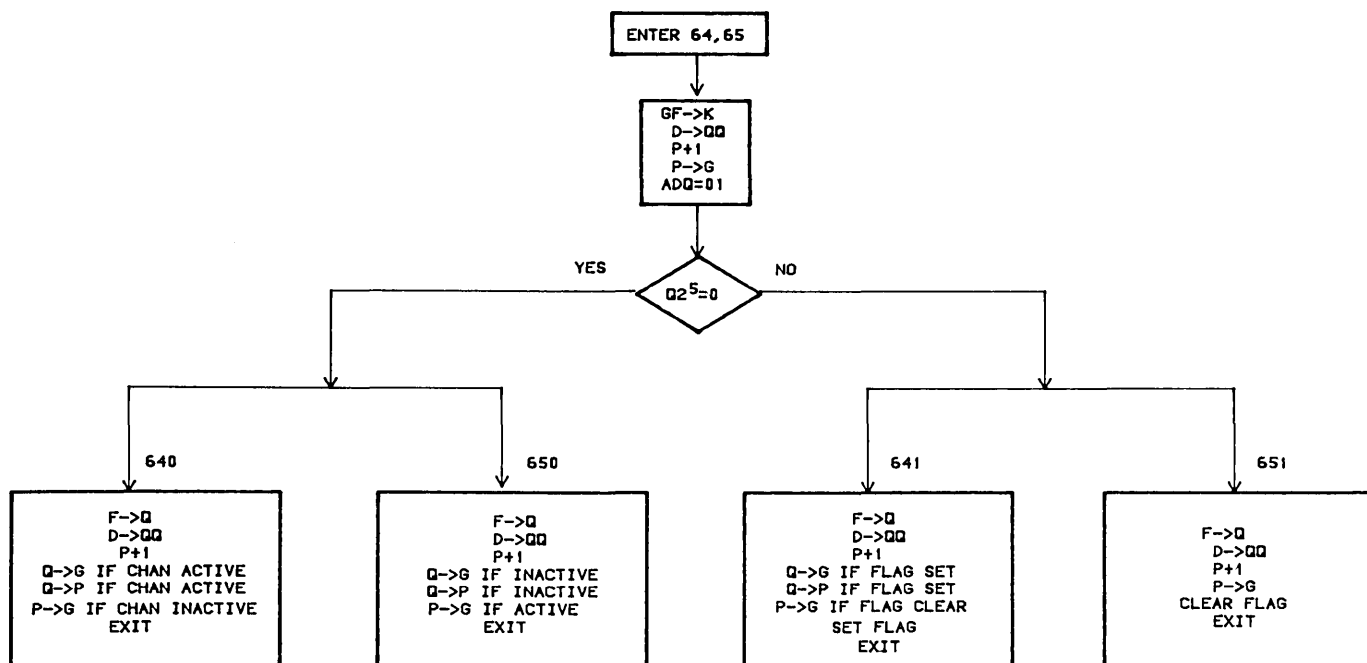
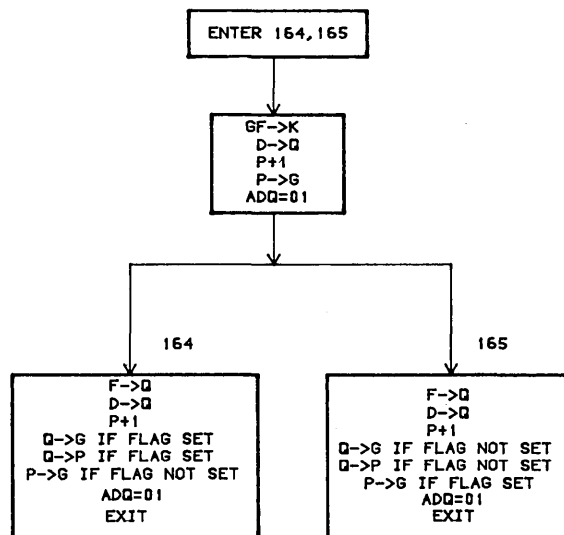


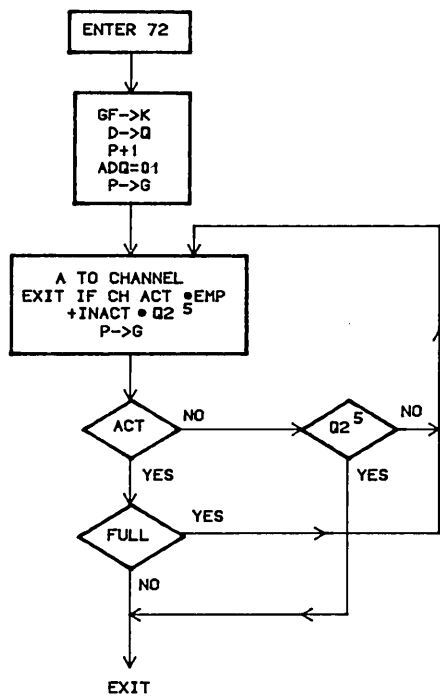
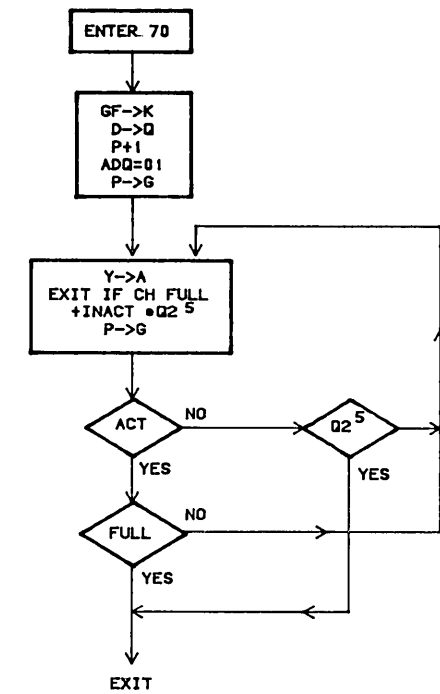
Figure B-28. Flowchart, PP Instructions 64 through 67

C1127



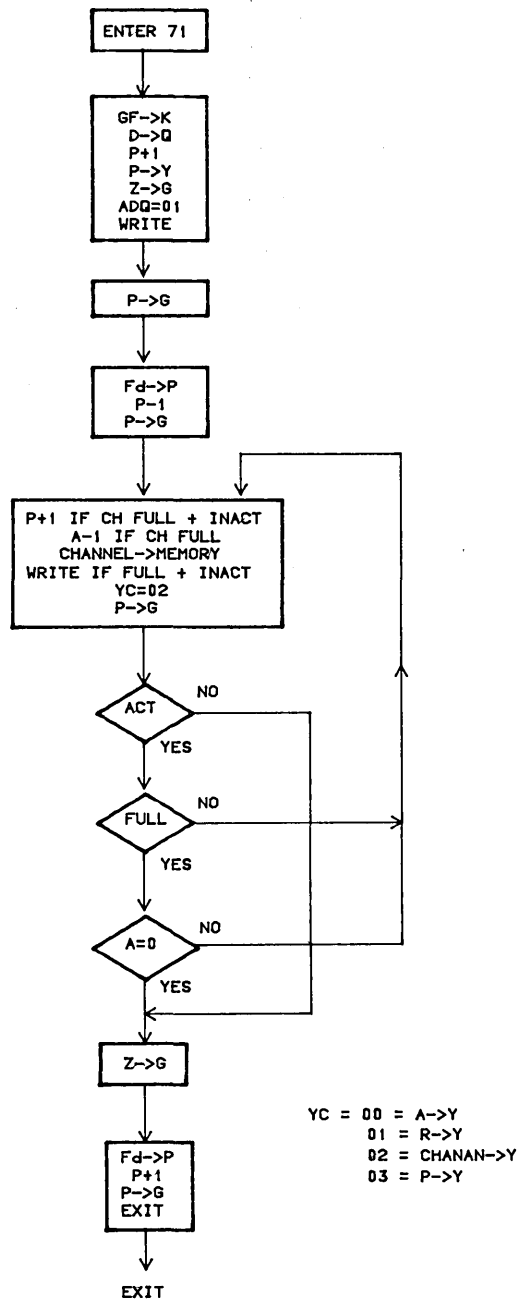
C1128

Figure B-29. Flowchart, PP Instructions 164 and 165



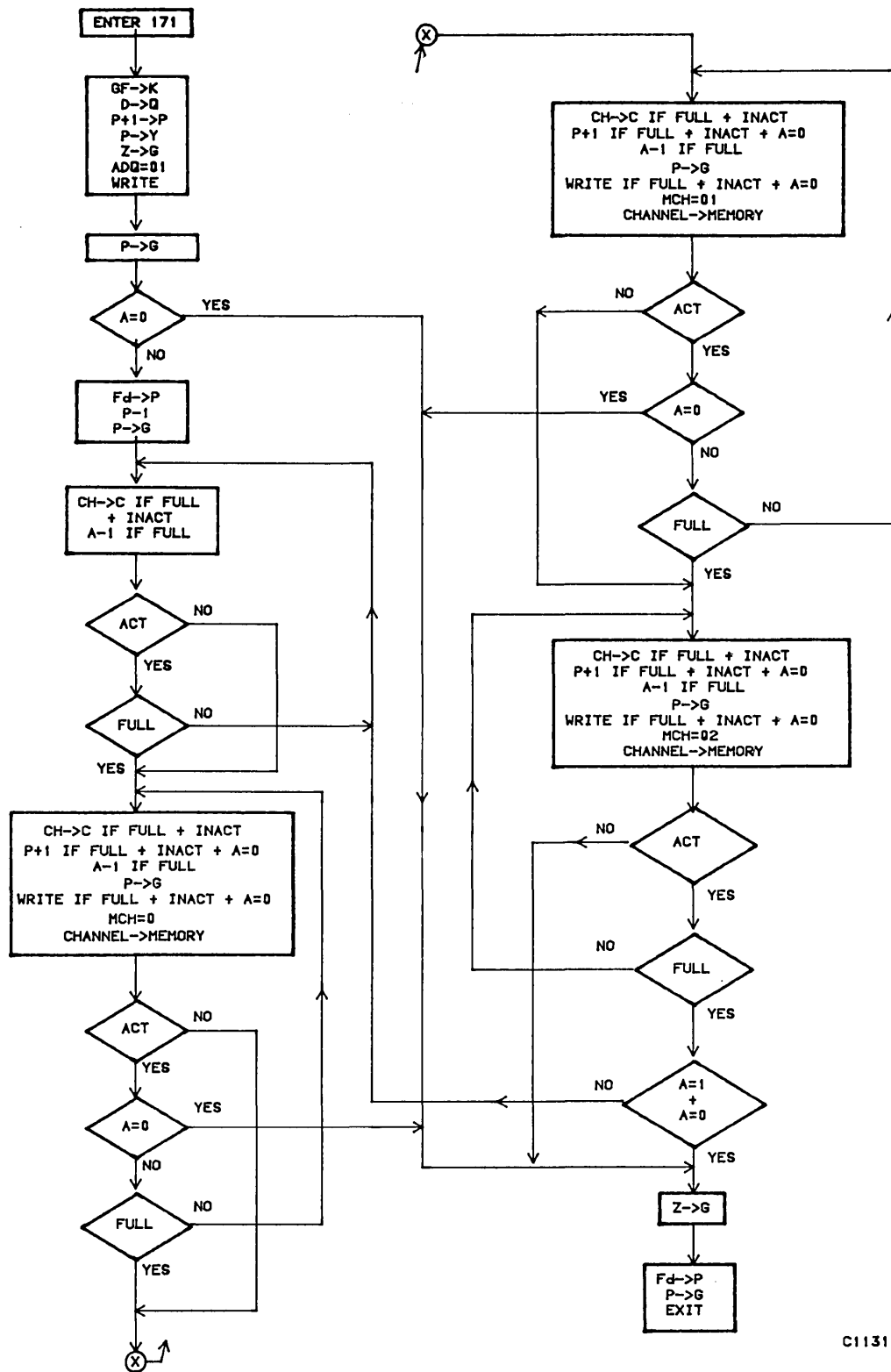
C1129

Figure B-30. Flowchart, PP Instructions 70 and 72



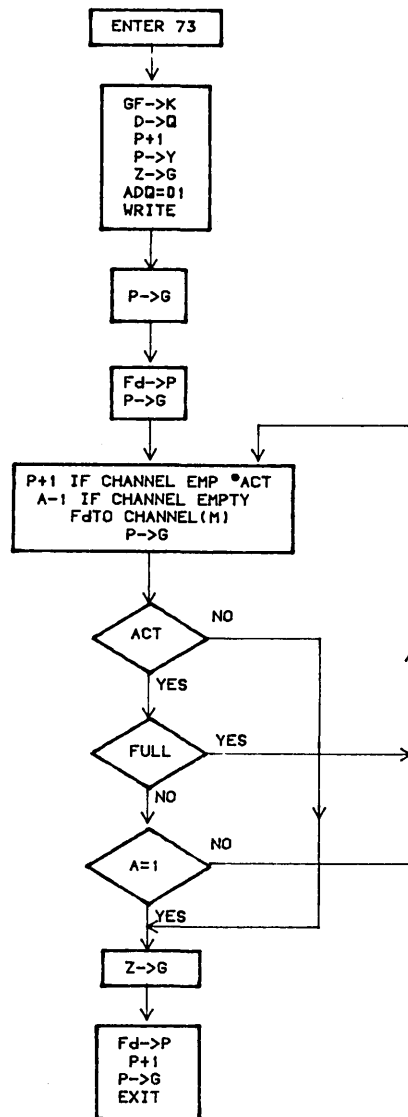
C1130

Figure B-31. Flowchart, PP Instruction 71



C1131

Figure B-32. Flowchart, PP Instruction 171



C1132

Figure B-33. Flowchart, PP Instruction 73

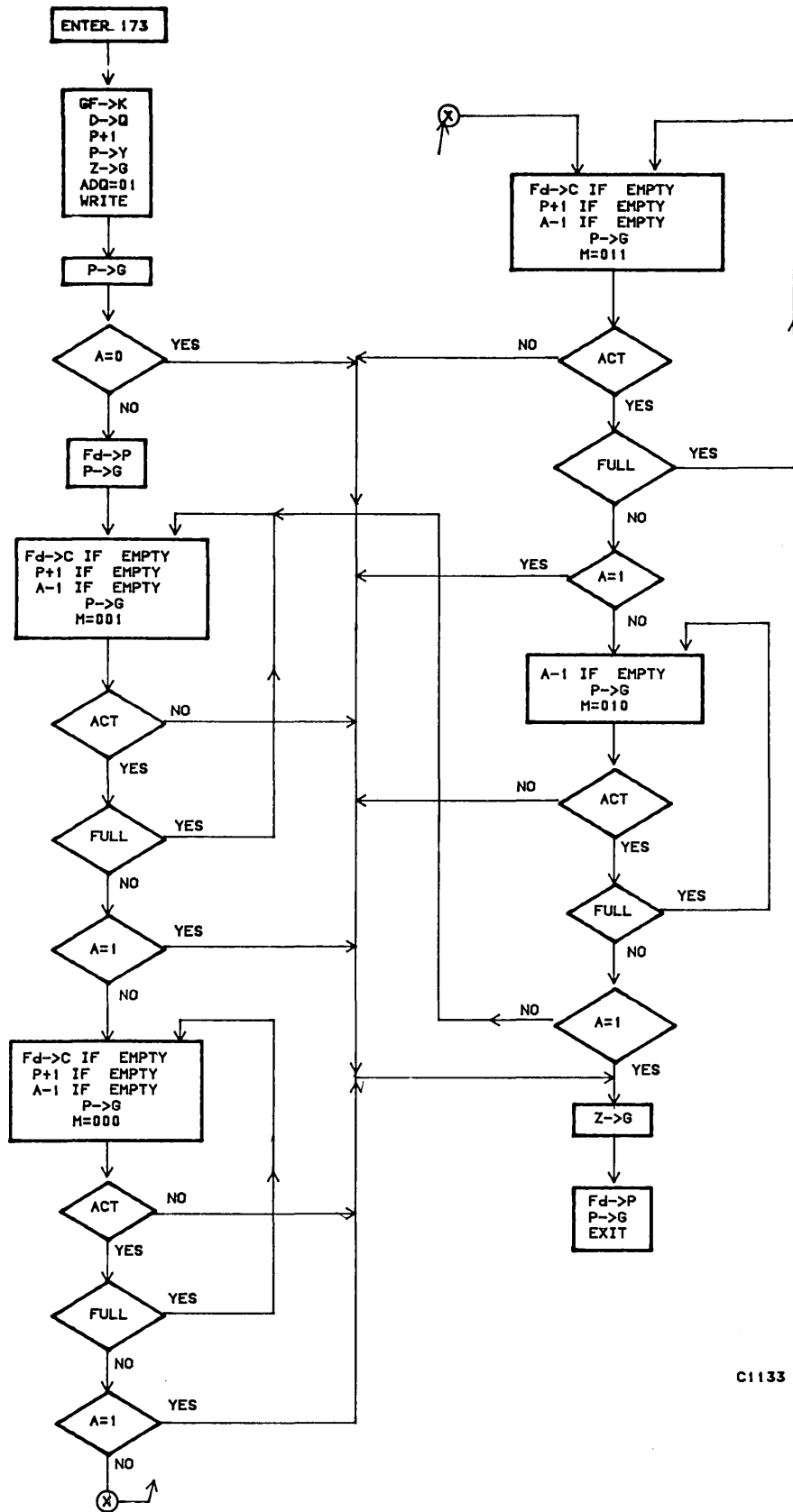
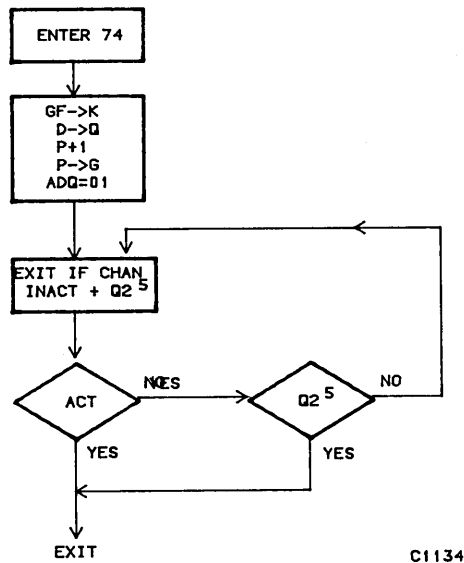
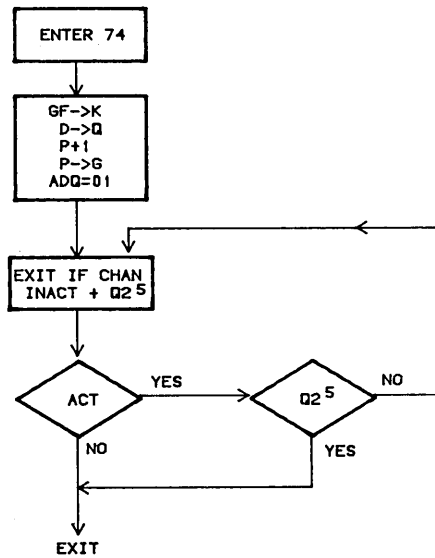
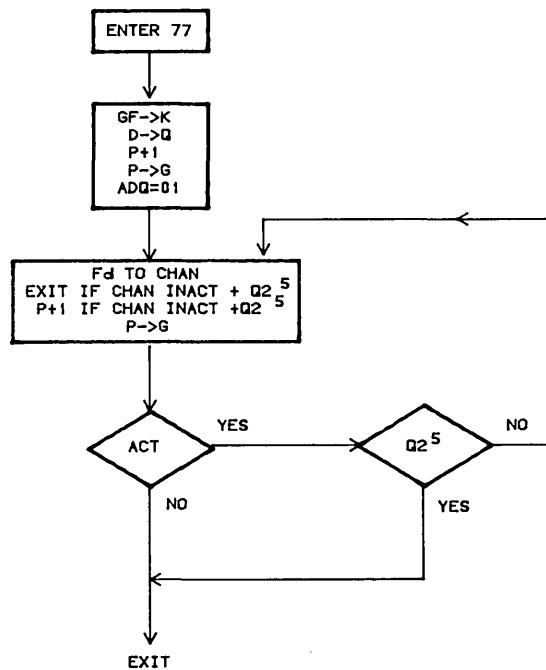
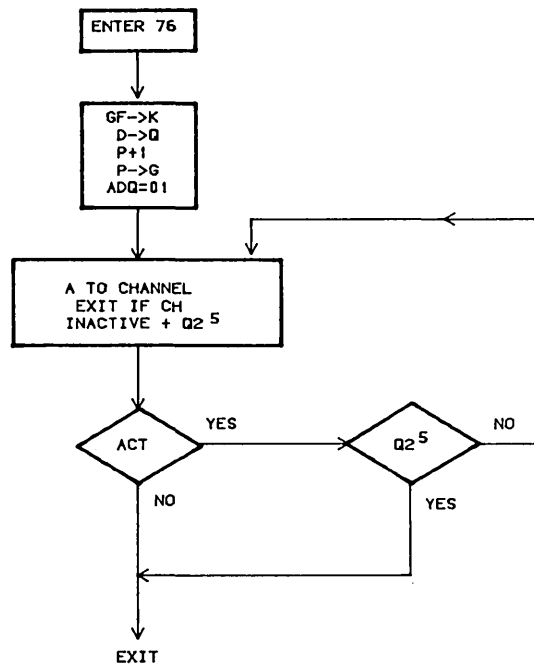


Figure B-34. Flowchart, PP Instruction 173



C1134

Figure B-35. Flowchart, PP Instructions 74 and 75



C1135

Figure B-36. Flowchart, PP Instructions 76 and 77

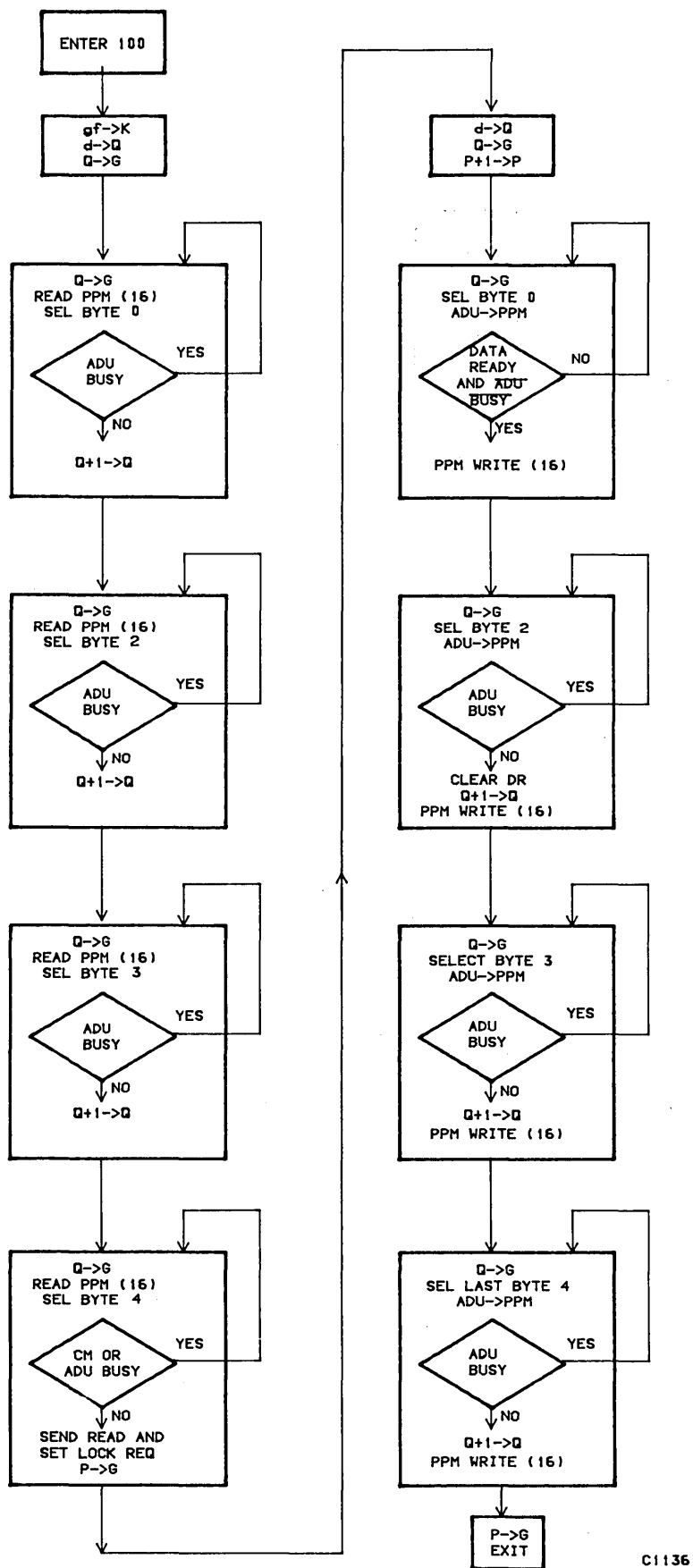
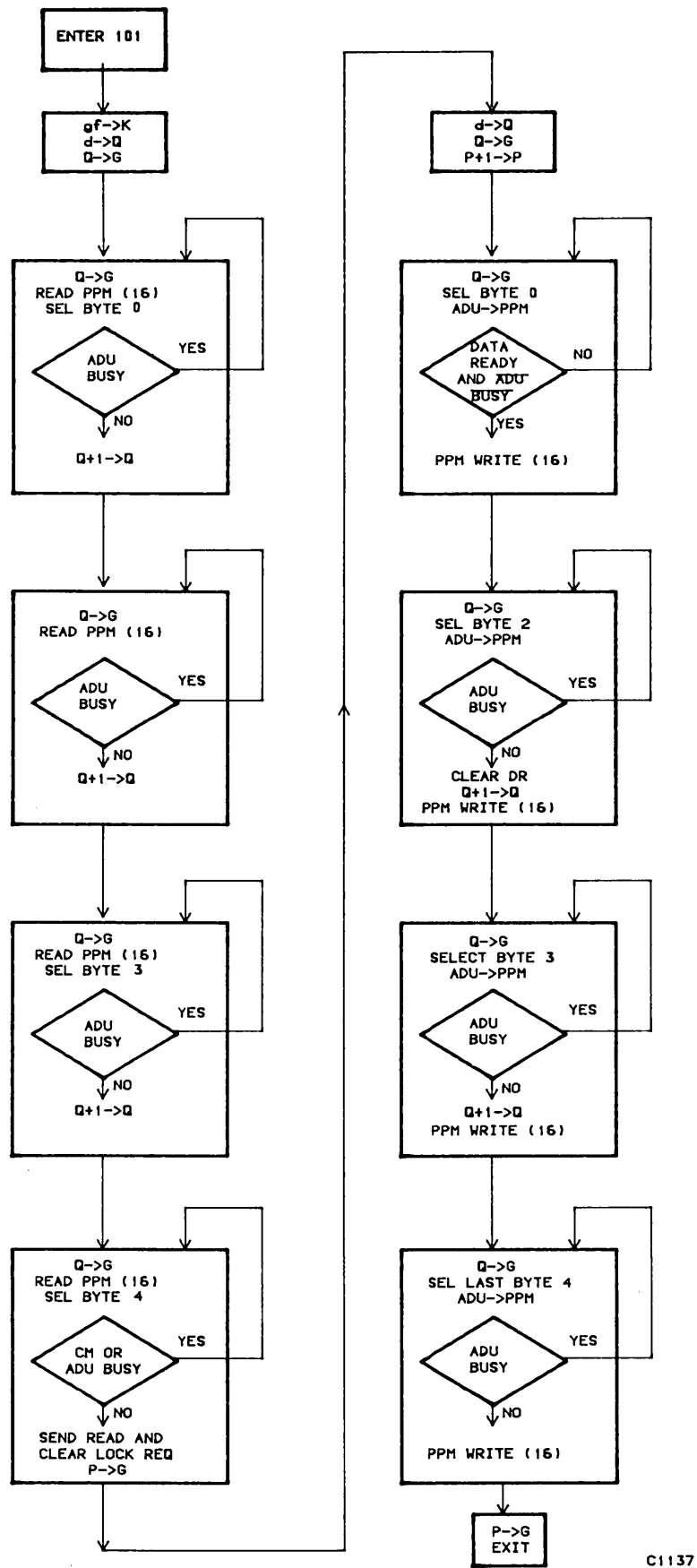


Figure B-37. Flowchart, PP Instruction 100



C1137

Figure B-38. Flowchart, PP Instruction 101

IOU INSTRUCTION TIMING

Execution times for IOU instructions are listed below. Times assume no conflicts occur. Timing notes refer to the end of the table. Execution times are in major cycles (1 major cycle equals 500 nanoseconds).

<u>Code</u>	<u>Description</u>	<u>Execution Time in Major Cycles</u>	<u>Timing Notes</u>
0000	Pass	1	
0001dm	Long jump to m+(d)	3	
0002dm	Return jump to m+(d)	4	
0003d	Unconditional jump d	1	
0005d	Non-zero jump d	1	
0006d	Plus jump d	1	
0007d	Minus jump d	1	
0010d	Shift (A) by d	1	
0011d	Logical difference (A) and d	1	
0012d	Logical product (A) and d	1	
0013d	Selective clear (A) by d	1	
0014d	Load d	1	
0015d	Load complement d	1	
0016d	Add (A)+d	1	
0017d	Subtract (A)-d	1	
0020dm	Load dm	2	
0021dm	Add (A)+dm	2	
0022dm	Logical product (A) and dm	2	
0023dm	Logical difference (A) and dm	2	
002400	Pass	1	
0024d	Load R from (d) and (d)+1	3	
002500	Pass	1	
0025d	Store R at (d) and (d)+1	4	
00260X	Exchange jump	-	1
00261X	Monitor exchange jump	-	1
00262X	Monitor exchange jump to MA	-	1
0027X	Keypoint	1	
0030d	Load (d)	2	
0031d	Add (A) + (d)	2	
0032d	Subtract (A)-(d)	2	
0033d	Logical difference (A) and (d)	2	
0034d	Store (d)	2	
0035d	Replace (A)+(d)	4	
0036d	Replace add one (d)	5	
0037d	Replace subtract one (d)	5	
0040d	Load ((d))	3	
0041d	Add (A)+((d))	3	
0042d	Subtract (A)-((d))	3	
0043d	Logical difference A and ((d))	3	
0044d	Store ((d))	3	

0045d	Replace add (A)+((d))	5	
0046d	Replace add one ((d))	6	
0047d	Replace subtract one ((d))	6	
0050dm	Load (m+(d))	4	
0051dm	Add (A)+(m+(d))	4	
0052dm	Subtract (A)-(m+(d))	4	
0053dm	Logical difference (A) and (m+(d))	4	
0054dm	Store (m+(d))	4	
0055dm	Replace add (A)+(m+(d))	6	
0056dm	Replace add one (m+(d))	7	
0057dm	Replace subtract one (m+(d))	7	
0060d	Central read from ((R)+(A)) to d	8	2
0061dm	Central read (d) words from ((R)+(A)) to m	7+5(d)	2,3
0062d	Central write to ((R)+(A)) from d	6	2
0063dm	Central write (d) words to ((R)+(A)) from m	6+5(d)	2,4
00640cm	Jump to m if channel c active	2	
00641cm	Jump if channel C flag set and set flag if clear	2	
00650cm	Jump to m if channel c inactive	2	
00651cm	Clear flag on channel c and jump to P+2	2	
00660cm	Jump to m if channel c full	2	
00661cm	Jump to m if channel c error flag set	2	
00670cm	Jump to m if channel c empty	2	
00671cm	Jump to m if channel c error flag clear	2	
00700c	Input to A from channel c when active and full	2	
00701c	Input to A from channel c if active and full	2	
0071Xcm	Input (A) words to m from channel c	-	5
00720c	Output from A on channel c when active and empty	2	
00721c	Output from A on channel c if active and empty	2	
0073Xcm	Output (A) words from m on channel c	-	5
00740c	Activate channel c when inactive	2	
00741c	Unconditionally activate channel c	2	
00750c	Deactivate channel c when channel active	2	
00751c	Unconditionally deactivate channel c	2	
00760c	Function (A) on channel c when inactive	2	

00761c	Function (A) on channel c	2	
	if channel inactive		
00770c m	Function m on channel c	2	
	when inactive		
00771c m	Function m on channel c	2	
	if inactive		
1000d	Central read and set lock	11	2
	from d to ((R)+(A))		
1001d	Central read and clear	11	2
	lock from d to ((R)+(A))		
1002	Pass	1	
1003	Pass	1	
1004	Pass	1	
1005	Pass	1	
1006	Pass	1	
1007	Pass	1	
1010	Pass	1	
1011	Pass	1	
1012	Pass	1	
1013	Pass	1	
1014	Pass	1	
1015	Pass	1	
1016	Pass	1	
1017	Pass	1	
1020	Pass	1	
1021	Pass	1	
1022d	Logical Product (A) and (d)	2	
1023d	Logical Product (A) and ((d))	3	
1024dm	Logical Product (A) and (m+(d))	4	
1025	Pass	1	
1026d	Interrupt processor on	2	2
	memory port d		
1027	Pass	1	
1030d	Load (d)	2	
1031d	Add (A)+(d)	2	
1032d	Subtract (A)-(d)	2	
1033d	Logical difference (A) and (d)	2	
1034d	Store (d)	2	
1035d	Replace (A)+(d)	4	
1036d	Replace add one (d)	5	
1037d	Replace subtract one (d)	5	
1040d	Load ((d))	3	
1041d	Add (A)+((d))	3	
1042d	Subtract (A)-((d))	3	
1043d	Logical difference (A) and ((d))	3	
1044d	Store ((d))	3	
1045d	Replace add (A)+((d))	5	
1046d	Replace add one (A)+((d))	6	
1047d	Replace subtract one ((d))	6	
1050dm	Load (m+(d))	4	

1051dm	Add (A)+(m+(d))	4	
1052dm	Subtract (A)-(m+(d))	4	
1053dm	Logical difference (A) and (m+(d))	4	
1054dm	Store (m+(d))	4	
1055dm	Replace add (A)+(m+(d))	6	
1056dm	Replace add one (m+(d))	7	
1057dm	Replace subtract one (m+(d))	7	
1060d	Central read from ((R)+(A)) to d	7	2
1061dm	Central read (d) words from ((R)+(A)) to m	7+4(d)	2,6
1062d	Central write to ((R)+(A)) from d	5	2
1063dm	Central write (d) words to ((R)+(A)) from m	6+4(d)	2,7
1064Xcm	Jump to m if channel c flag set	2	
1065Xcm	Jump to m if channel c flag clear	2	
1066d	Pass	1	
1067d	Pass	1	
1070d	Pass	1	
1071Xcm	Input (A) words to m from channel c packed	-	5,8
1072d	Pass	1	
1073Xcm	Output (A) words from m on channel c packed	-	5,8
1074d	Pass	1	
1075d	Pass	1	
1076d	Pass	1	
1077d	Pass	1	

NOTES

1. Two major cycles plus the waiting time for the Exchange Accept signal from the CPU.
2. No PP conflicts (CM busy flip flop set) and no CM conflicts.
3. Five major cycles for instruction setup. Five major cycles per CM word. Two major cycles for instruction exit.
4. Four major cycles for instruction setup. Five major cycles per CM word. Two major cycles for instruction exit.
5. Three major cycles for instruction setup. Maximum rate of one major cycle per channel word. Two major cycles for instruction exit.
6. Five major cycles for instruction setup. Four major cycles per CM word. Two major cycles for instruction exit.
7. Four major cycles for instruction setup. Four major cycles per CM word. Two major cycles for instruction exit.

8. If the channel word count (A) is zero initially, the instruction takes four major cycles.

IOU INSTRUCTIONS

The bracketed expression following each instruction code is the COMPASS Assembler mnemonic opcode.

LOAD AND STORE

The load and store instructions transfer 6-bit, 12-bit, 16-bit, and 18-bit quantities between A register and PPM.

Load d 0014d [LDN d]

Load d enters a copy of the d-field, considered a 6-bit positive integer, into bits 58 through 63 of the A register. Bits 46 through 57 of A register are cleared.

Load Complement d 0015 d [LCN d]

Load complement d enters a complemented copy of the d-field into bits 58 through 63 of the A register. Bits 46 through 57 of A register are set to one, and bits 58 through 63 are bit-by-bit complements of the corresponding bits in the d-field.

Load dm 0020 d m [LDC dm]

Load dm clears the A register and enters an 18-bit operand consisting of the 6-bit d-field and the 12-bit m-field into bits 46 through 51 and bits 52 through 63, respectively, of the A register.

Load (d) 0030 d [LDD d]

Load (d) clears the A register and enters a 12-bit quantity, treated as a 12-bit positive integer, from bits 52 through 63 of location d into bits 52 through 63 of A register. Bits 46 through 51 of A register are cleared.

Load (d) Long 1030 d [LDDL d]

Load (d) long clears the A register and enters a 16-bit quantity, treated as a 16-bit positive integer, from location d into bits 48 through 63 of A register. Bits 46 through 47 of A register are cleared.

Store (d) 0034 d [STD d]

Store (d) stores the 12-bit quantity in bits 52 through 63 of the A register into location d. Bits 48 through 51 of location d are cleared. The contents of A register are not altered.

Store (d) Long 1034 d [STD L d]

Store (d) long stores the 16-bit quantity in bits 48 through 63 of the A register into location d. The contents of A register are not altered.

Load ((d)) 0040 d [LDI d]

Load ((d)) clears A register and loads bits 52 through 63 of ((d)) into bits 52 through 63 of A register. Bits 46 through 51 of A register are cleared.

Load ((d)) Long 1040 d [LDIL d]

Load ((d)) long clears A register and loads ((d)) into bits 48 through 63 of A register. Bits 46 and 47 of A register are cleared.

Store ((d)) 0044 d [STI d]

Store ((d)) stores bits 52 through 63 of the A register into bits 52 through 63 of the storage location specified by the contents of location d. Bits 48 through 51 of ((d)) are cleared. The contents of A register are not altered.

Store ((d)) Long 1044 d [STIL d]

Store ((d)) long stores bits 48 through 63 of the A register into the storage location specified by the contents of location d. The contents of A register are not altered.

Load (m+(d)) 0050 d m [LDM m,d]

Load (m+(d)) clears the A register and enters bits 52 through 63 of a 12-bit operand from storage into bits 52 through 63 of A register. The address for the operand is formed by adding bits 52 through 63 of the m-field and bits 52 through 63 of the contents of location d in a 12-bit ones complement mode. If the d-field is zero, the operand address is given by m. The operand enters A register as a 12-bit positive integer and bits 46 through 51 of A register are cleared.

Load (m+(d)) Long 1050 d m [LDML m,d]

Load (M+(d)) long clears the A register and enters a 16-bit operand from storage into bits 48 through 63 of A register. The address for the operand is formed by adding bits 52 through 63 of the m-field and bits 52 through 63 of

the contents of location d in a 12-bit ones complement mode. If the d-field is zero, the operand address is given by m. The operand enters A register as a 16-bit positive integer and bits 46 and 47 of A register are cleared.

Store (m+(d)) 0054 d m [STM m,d]

Store (m+(d)) stores bits 52 through 63 of the A register into bits 52 through 63 of storage. Bits 48 through 51 of (m+(d)) are cleared. The storage address is formed by adding bits 52 through 63 of the m-field and bits 52 through 63 of the contents of location d in a 12-bit ones complement mode. If the d-field is zero, then the storage address is given by m. The contents of A register are not altered.

Store (m+(d)) Long 1054 d m [STML m,d]

Store (m+(d)) long stores bits 48 through 63 of the A register into storage. The storage address is formed by adding bits 52 through 63 of the m-field and bits 52 through 63 of the contents of location d in a 12-bit ones complement mode. If the d-field is zero, then the storage address is given by m. The contents of A register are not altered.

ARITHMETIC

The arithmetic instructions perform integer arithmetic with the contents of A as one operand and the other operand as specified by the instruction. The result replaces the original contents of A. The operands are considered ones complement integers and the arithmetic is performed in ones complement logic.

Add d 0016 d [ADN d]

Add d adds the d-field, considered a 6-bit positive quantity, to the current contents of the A register. The result remains in A register. The addition is in an 18-bit ones complement logic. An 18-bit operand is formed from the d-field by adding 12 highest order zero bits.

Subtract d 0017 d [SBN d]

Subtract d subtracts the d-field, considered a 6-bit positive quantity, from the current contents of the A register. The result remains in A register. An 18-bit operand is formed from the d-field. This operand consists of 12 highest order one bits, and six lowest order bits which are bit-by-bit complements of the corresponding bits in the d-field. This 18-bit operand adds to the original contents of A register in an 18-bit ones complement logic.

Add dm 0021 d m [ADC dm]

Add dm adds an 18-bit operand consisting of the d- and m-fields to the current contents of the A register. The result remains in A register. The addition is in an 18-bit ones complement logic. The d-field forms the highest order 6 bits, and the m-field completes the lowest order 12 bits.

Add (d) 0031 d [ADD d]

Add (d) adds bits 52 through 63 of location d, considered a 12-bit positive integer, to the current contents of the A register. The result remains in A register. The addition is in an 18-bit ones complement logic. An 18-bit operand is formed from bits 52 through 63 of location d by adding six highest-order zero bits.

Add (d) Long 1031 d [ADDL d]

Add (d) long adds the contents of location d, considered a 16-bit positive integer, to the current contents of the A register. The result remains in A register. The addition is in an 18-bit ones complement logic. An 18-bit operand is formed from location d by adding two highest order zero bits.

Subtract (d) 0032 d [SBD d]

Subtract (d) subtracts bits 52 through 63 of location d, considered a 12-bit positive integer from the current contents of the A register. The result remains in A register. The operation adds the complement of location d to the contents of A register in an 18-bit ones complement logic. An 18-bit operand is formed from location d. This operand consists of six highest order one bits, and twelve lowest order bits which are bit-by-bit complements of the corresponding bits in location d.

Subtract (d) Long 1032 d [SBDL d]

Subtract (d) long subtracts the contents of location d, considered a 16-bit positive integer from the current contents of the A register. The result remains in A register. The operation adds the complement of location d to the contents of A register in an 18-bits ones complement logic. An 18-bit operand is formed from location d. This operand consists of two highest order one bits, and 16 lowest order bits which are bit-by-bit complements of the corresponding bits in location d.

Add ((d)) 0041 d [ADI d]

Add ((d)) adds a 12-bit operand considered a 12-bit positive integer, from bits 52 through 63 of storage to the current contents of the A register. The address for the operand is in location d. The result remains in A register. The addition is in an 18-bit ones complement logic. An 18-bit operand is formed from the 12-bit operand by adding six highest order zero bits.

Add ((d)) Long 1041 d [ADIL d]

Add ((d)) long adds a 16-bit operand, considered a 16-bit positive integer to the current contents of the A register. The address for the operand is in location d. The result remains in A register. The addition is in an 18-bit ones complement logic. An 18-bit operand is formed from the 16-bit operand by adding two highest order zero bits.

Subtract ((d)) 0042 d [SBI d]

Subtract ((d)) reads an operand from bits 52 through 63 of storage and subtracts it from the current contents of the A register. The result remains in A register. The address for the operand is in location d. The complement of the operand is added to the contents of A register in an 18-bit ones complement logic. An 18-bit operand for the addition is formed from the 12-bit storage operand by forcing each of the highest order six bits to a one. The lowest order 12 bits are the bit-by-bit complement of the storage operand values.

Subtract ((d)) Long 1042 d [SBIL d]

Subtract ((d)) long reads an operand from storage and subtracts it from the current contents of the A register. The result remains in A register. The address for the operand is in location d. The complement of the operand is added to the contents of A register in an 18-bit ones complement logic. An 18-bit operand for the addition is formed from the 16-bit storage operand by forcing each of the highest order two bits to a one. The lowest order 16 bits are the bit-by-bit complement of the storage operand values.

Add (m+(d)) 0051 d m [ADM m,d]

Add (m+(d)) reads an operand from bits 52 through 63 of storage and adds it to the current contents of the A register. The result remains in A register. The addition is in an 18-bit ones complement logic. An 18-bit operand is formed from the 12-bit storage operand by adding six highest order zero bits. The storage address for the operand is formed by adding bits 52 through 63 of the m-field and bits 52 through 63 of the contents of location d in a 12-bit ones complement logic. If the d-field is zero, then the storage address is given by m.

Add (m+(d)) Long 1051 d m [ADML m,d]

Add (m+(d)) long reads an operand from storage and adds it to the current contents of the A register. The result remains in A register. The addition is in an 18-bit ones complement logic. An 18-bit operand is formed from the 16-bit storage operand by adding two highest order zero bits. The storage address for the operand is formed by adding bits 52 through 63 of the m-field and bits 52 through 63 of the contents of location d in a 12-bit ones complement logic. If the d-field is zero, then the storage address is given by m.

Subtract (m+(d)) 0052 d m [SBM m,d]

Subtract (m+(d)) reads an operand from bits 52 through 63 of storage and subtracts it from the current contents of the A register. The result remains in A register. The complement of the storage operand is added to the contents of A register in the 18-bit ones complement logic. An 18-bit operand is formed from the 12-bit storage operand by forcing each of the highest order six bits to a one. The lowest order 12 bits are the bit-by-bit complement of the storage operand. The storage address for the operand is formed by adding bits 52 through 63 of the m-field and bits 52 through 63 of location d in a 12-bit ones complement logic. If the d-field is zero, then the storage address is given by m.

Subtract (m+(d)) Long 1052 d m [SBML m,d]

Subtract (m+(d)) long reads an operand from storage and subtracts it from the current contents of the A register. The result remains in A register. The complement of the storage operand is added to the contents of A register in 18-bit ones complement logic. An 18-bit operand is formed from the 16-bit storage operand by forcing each of the highest order 2 bits to a one. The lowest order 16 bits are the bit-by-bit complement of the storage operand. The storage address for the operand is formed by adding bits 52 through 63 of the m-field and bit 52 through 63 of the contents of location d in a 12-bit ones complement logic. If the d-field is zero, then the storage address is given by m.

LOGICAL

The logical instructions perform operations with the contents of A register as one operand and the other operand specified by the instruction. The result replaces the original contents of A register.

Shift d 0010 d [SHN d]

Shift d shifts the contents of the A register either to the right open-ended or to the left circularly as specified by the d-field. The d-field is treated as a 6-bit ones complement number. If the highest order bit in the d-field is zero, then the contents of A register shift circularly to the left by the number of bit positions indicated in the value of the d-field. If the highest order bit in the d-field is one, the contents of A register shift open-ended to the right by the complement of the value of the d-field.

In a left circular shift, the contents of A register shift one bit at a time. In each shift, the lowest order bit position in the register is filled by the bit previously held in the highest order bit position. Bits are not lost in this process but are repositioned toward the highest order positions. A d-field of zero causes no shift. A d-field in the range of 18 through 31 (decimal) causes a complete shift in the register, resulting in a shift of d-18 (decimal).

In a right open-ended shift, the contents of A register shift one bit position at a time toward the lower-order bit positions in the register. The highest-order bit position in A register is filled with a zero value as each shift occurs. The lowest order bit in A register is discarded as each shift occurs. A maximum of 31 (decimal) shift counts may be used. For all shift counts larger than 17 (decimal), the final A register value is 000000 (octal). A d-field of 77 (octal) causes no shift.

Logical Difference d 0011 d [LMN d]

Logical difference d performs the logical difference function of d, considered a 6-bit positive integer, and bits 58 through 63 of A register. Bits 46 through 57 of A register are not altered.

The logical difference function forms the bit-by-bit logical operation on two operands as follows:

operand 1	0 0 1 1
operand 2	<u>0 1 0 1</u>
result	0 1 1 0

Logical Product d 0012 d [LPN d]

Logical product d performs the logical product function of d, considered a 6-bit positive integer, and bits 58 through 63 of A register. Bits 46 through 57 of A register are cleared.

The logical product function forms the bit-by-bit logical operation on two operands as follows:

operand 1	0 0 1 1
operand 2	<u>0 1 0 1</u>
result	0 0 0 1

Selective Clear d 0013 d [SCN d]

Selective clear d clears any or all of bits 58 through 63 of A register if they are one. Bits 46 through 57 of A register are not altered.

Logical Product dm 0022 d m [LPC dm]

Logical product dm forms the logical product of the A register contents and an 18-bit operand consisting of the d- and m-fields. The result remains in A register. The d-field forms the highest order 6-bits, and the m-field completes the lowest order 12-bits of the 18-bit operand. The logical product is formed as described in the truth table given for the 0012 instruction above.

Logical Product (d) Long 1022 d [LPDL d]

Logical product (d) long forms, in the A register, the logical product of the contents of locations d, considered a 16-bit quantity, and the original contents of A register. Bits 46 and 47 of A register are cleared by this operation. The logical product is formed as described in the truth table given for the 0012 instruction.

Logical Product ((d)) Long 1023 d [LPIL d]

Logical product ((d)) long forms, in the A register, the logical product of a 16-bit operand read from storage and the original contents of the A register. The storage address for the operand is in location d. Bits 46 and 47 of A register are cleared by this operation. The logical product is formed as described in the truth table given for the 0012 instruction.

Logical Product (m+(d)) Long 1024 d m [LPML m,d]

Logical product (m+(d)) long forms, in the A register, the logical product of a 16-bit operand read from storage and the original contents of the A register. The address for the operand is formed by adding bits 52 through 63 of the m-field and bits 52 through 63 of the contents of location d in a 12-bit ones complement logic. If the d-field is zero, then the address of the operand is given by m. Bits 46 and 47 of A register are cleared by the operation. The logical product is formed as described in the truth table given for the 0012 instruction.

Logical Difference dm 0023 d m [LMC dm]

Logical difference dm forms, in the A register, the logical difference of the A register contents and an 18-bit operand consisting of the d- and m-fields. The d-field forms the highest order 6 bits, and the m-field completes the lowest order 12 bits of the 18-bit operand. The logical difference is formed according to the truth table given for the 0011 instruction above.

Logical Difference (d) 0033 d [LMD d]

Logical difference (d) forms, in the A register, the logical difference of bits 52 through 63 of the contents of location d, considered a 12-bit positive quantity and the original contents of A register. The highest order 6 bits of A register are not affected by this operation. The logical difference is formed according to the truth table given for the 0011 instruction.

Logical Difference (d) Long 1033 d [LMDL d]

Logical difference (d) long forms, in the A register, the logical difference of the contents of location d, considered a 16-bit positive quantity, and the original contents of A register. Bits 46 and 47 of A register are not affected by this operation. The logical difference is formed according to the truth table given for the 0011 instruction.

Logical Difference ((d)) 0043 d [LMI d]

Logical difference ((d)) forms the logical difference of bits 52 through 63 of an operand read from storage and the original contents of the A register. Bits 46 through 51 of A register are not affected by this operation. The storage address for the operand is in location d. The logical difference is formed according to the truth table given for the 0011 instruction.

Logical Difference ((d)) Long 1043 d [LMIL d]

Logical difference ((d)) long forms the logical difference of an operand read from storage and the original contents of the A register. Bits 46 and 47 of A register are not affected by this operation. The storage address for the operand is in location d. The logical difference is formed according to the truth table given for the 0011 instruction.

Logical Difference (m+(d)) 0053 d m [LMM m,d]

Logical difference (m+(d)) forms the logical difference of bits 52 through 63 of an operand read from storage and the original contents of the A register. Bits 46 through 51 of A register are not affected by this operation. The address for the operand is formed by adding bits 52 through 63 of the m-field and bits 52 through 63 of the contents of location d in 12-bit ones complement logic. If the d-field is zero, the address of the operand is given by m. The logical difference is formed according to the truth table given for the 0011 instruction.

Logical Difference (m+(d)) Long 1053 d m [LMML m,d]

Logical difference (m+(d)) long forms the logical difference of an operand read from storage and the original contents of the A register. Bits 46 and 47 of A register are not affected by this operation. The address for the operand is formed by adding bits 52 through 63 of the m-field and bits 52 through 63 of the contents of location d in 12-bit ones complement logic. If the d-field is zero, the address of the operand is given by m. The logical difference is formed according to the truth table given for the 0011 instruction.

REPLACE

The replace instructions are similar to the arithmetic instructions in that they perform integer arithmetic with the contents of A register as one operand and second operand whose location is specified by the instruction.

The result replaces the original contents of A register and the contents of the other operand. The operands are considered ones complement integers and the arithmetic is performed in ones complement logic.

Replace Add (d) 0035 d [RAD d]

Replace add (d) adds bits 52 through 63 of the contents of location d, considered a 12-bit positive integer, to the current contents of the A register. The result remains in A register and is stored in location d. The addition is in an 18-bit ones complement logic. An 18-bit operand is formed from the contents of location d by adding six highest order zero bits. The 16-bit result stored in location d is the lowest order 12 bits of the resulting 18-bit sum with four highest order zero bits added. The value in A register is not necessarily equal to the result in location d.

Replace Add (d) Long 1035 d [RADL d]

Replace add (d) long adds the contents of location d, considered a 16-bit positive integer, to the current contents of the A register. The result remains in A register and is stored in location d. The addition is in 18-bit ones complement logic. An 18-bit operand is formed from the contents of location d by adding two highest order zero bits. The result stored in location d is the lowest order 16 bits of the resulting 18-bit sum. The value in A register is not necessarily equal to the quantity in location d.

Add One (d) 0036 d [AOD d]

Add one (d) increases the contents of bits 52 through 63 of location d by one count. Conceptually, a value of plus one is entered into the A register. Bits 52 through 63 are then read from storage and added to the A register in an 18-bit ones complement logic. Bits 52 through 63 of location d are considered a 12-bit positive integer. An 18-bit operand is formed from this quantity by adding six highest order zero bits. The result remains in A register. Bits 52 through 63 of the resulting sum have four highest order zero bits added and are stored in location d. The value in A register is not necessarily equal to the quantity in location d.

Add One (d) Long 1036 d [AODL d]

Add one (d) long increases the contents of location d by one count. Conceptually, a value of plus one is entered into the A register. The contents of location d are then read from storage and added to the A register in 18-bit ones complement logic. The contents of location d are considered a 16-bit positive quantity. An 18-bit operand is formed from this quantity by adding two highest order zero bits. The result remains in A register. Bits 48 through 63 of the resulting sum in the A register are stored in location d. The value in A register is not necessarily equal to the quantity in location d.

Subtract One (d) 0037 d [SOD d]

Subtract one (d) decreases the contents of bits 52 through 63 of location d by one count. Conceptually, a value of minus one is entered into the A register. Bits 52 through 63 of the contents of location d are then read from storage and added to the A register in 18-bit ones complement logic. Bits 52 through 63 of location d are considered a 12-bit positive integer. An 18-bit

operand is formed from this quantity by adding six highest order zero bits. The result remains in A register. Bits 52 through 63 of the resulting sum in the A register have four highest order zero bits added and are stored in location d. The value in A register is not necessarily equal to the quantity in location d.

Subtract One (d) Long 1037 d [SODL d]

Subtract one (d) long decreases the contents of location d by one. Conceptually, a value of minus one is entered into the A register. The contents of location d are then read from storage and added to the A register in 18-bit ones complement logic. The contents of location d are considered a 16-bit positive integer. An 18-bit operand is formed from this quantity by adding two highest order zero bits. The result remains in A register. Bits 48 through 63 of the resulting sum in the A register are stored in location d. The value in A register is not necessarily equal to the quantity in location d.

Replace Add ((d)) 0045 d [RAI d]

Replace add ((d)) reads bits 52 through 63 of an operand from storage and adds it to the current contents of the A register. The result remains in A register and is also stored in the same memory location from which the operand was read. The addition is in 18-bit ones complement logic. An 18-bit operand is formed from the 12-bit storage operand by adding six highest order bits. Bits 52 through 63 of the resulting sum in the A register have four highest order zero bits added and returned to storage. The value in A register is not necessarily equal to the quantity in location (d). The storage address for reading the operand and storing the result is in location d.

Replace Add ((d)) Long 1045 d [RAIL d]

Replace add ((d)) long reads an operand from storage and adds it to the current contents of the A register. The result remains in A register and is also stored in the same memory location from which the operand was read. The addition is in 18-bit ones complement logic. An 18-bit operand is formed from the 16-bit storage operand by adding two highest order zero bits. Bits 48 through 63 of the resulting sum in the A register are returned to storage. The value in A register is not necessarily equal to the quantity in location (d). The storage address for reading the operand and storing the result is in location d.

Add One ((d)) 0046 d [AOI d]

Add One ((d)) increases bits 52 through 63 of an operand in storage by one. Conceptually, a value of plus one is entered into the A register. Bits 52 through 63 of the operand in storage are then read and added to the contents of A register in an 18-bit ones complement logic. The operand is treated as a 12-bit positive integer. An 18-bit operand is formed from the 12-bit storage operand by adding six highest order zero bits. The result remains in A. Bits 52 through 63 have four highest order zero bits added and are returned to location (d). The value in A register is not necessarily equal to the quantity

in location (d). The storage address for reading the operand and storing the result is in location d.

Add One ((d)) Long 1046 d [AOIL d]

Add one ((d)) long increases the value of an operand in storage by one. Conceptually, a value of plus one is entered into the A register. The storage operand is then read and added to the contents of A register in an 18-bit ones complement logic. The operand is treated as a 16-bit positive integer. An 18-bit operand is formed from the 16-bit storage operand by adding two highest-order zero bits. The result remains in A register, and bits 48 through 63 are returned to location (d). The value in A register is not necessarily equal to the quantity in location (d). The storage address for reading the operand and storing the result is in location d.

Subtract One ((d)) 0047 d [SOI d]

Subtract One ((d)) reads bits 52 through 63 of an operand from storage, decreases its value by one and returns bits 52 through 63 of the result to the same storage location. Conceptually, a value of minus one is entered into the A register. Bits 52 through 63 of the operand in storage are then read and added to the contents of A register in an 18-bit ones complement logic. The operand is treated as a 12-bit positive integer. An 18-bit operand is formed from the 12-bit storage operand by adding six highest order zero bits. The result remains in A register, and bits 52 through 63 have four highest order zero bits added and are returned to location (d). The value in A register is not necessarily equal to the quantity in location (d). The storage address for reading the operand and storing the result is in location d.

Subtract One ((d)) Long 1047 d [SOIL d]

Subtract one ((d)) long reads an operand from storage, decreases its value by one, and returns the result to the same storage location. Conceptually, a value of minus one is entered into the A register. The operand then reads from storage and adds to the contents of A register in an 18-bit ones complement logic. The operand is treated as a 16-bit positive integer. An 18-bit operand is formed from the 16-bit storage operand by adding two highest order zero bits. The result remains in A register, and bits 48 through 63 are returned to location (d). The value in A register is not necessarily equal to the quantity in location (d). The storage address for reading the operand and storing the result is in location d.

Replace Add (m+(d)) 0055 d m [RAM m,d]

Replace add (m+(d)) reads bits 52 through 63 of an operand from storage and adds it to the current contents of the A register. The result remains in A register, and bits 52 through 63 are stored in the same memory location from which the operand was read. The addition is in an 18-bit ones complement logic. An 18-bit operand is formed from the 12-bit storage operand by adding six highest order zero bits. Bits 52 through 63 of the result in A register have four highest order zero bits added and are returned to storage. The value

in A register is not necessarily equal to the quantity returned to storage. The storage address for reading the operand and storing the result is formed by adding the m-field to the contents of location d in a 12-bit ones complement logic. If the d-field is zero, then the storage address is given by m.

Replace Add (m+(d)) Long 1055 d m [RAML m,d]

Replace add (m+(d)) long reads an operand from storage and adds it to the current contents of the A register. The result remains in A register, and bits 48 through 63 are stored in the same memory location from which the operand was read. The addition is in 18-bit ones complement logic. An 18-bit operand is formed from the 16-bit storage operand by adding two highest order zero bits. Bits 48 through 63 of the result in A register are returned to storage. The value in A register is not necessarily equal to the quantity returned to storage. The storage address for reading the operand and storing the result is formed by adding bits 52 through 63 of the m-field to bits 52 through 63 of location d in 12-bit ones complement logic. If the d-field is zero, then the storage address is given by m.

Add One (m+(d)) 0056 d m [AOM m,d]

Add one (m+(d)) reads bits 52 through 63 of an operand from storage, increases its value by one, and returns bits 52 through 63 of the result to the same storage location. Conceptually, a value of plus one is entered into the A register. The operand then reads from storage and adds to the contents of A register in 18-bit ones complement logic. The operand is treated as a 12-bit positive integer in this process. An 18-bit operand is formed from the 12-bit storage operand by adding six highest order zero bits. The result remains in A register, and bits 52 through 63 have four highest order zero bits added and are returned to storage. The value in A register is not necessarily equal to the quantity returned to storage. The storage address for reading the operand and storing the result is formed by adding bits 52 through 63 of the m-field to bits 52 through 63 of location d in a 12-bit ones complement logic. If the d-field is zero, then the storage address is given by m.

Add One (m+(d)) Long 1056 d m [AOML m,d]

Add one (m+(d)) long reads an operand from storage, increases its value by one, and returns the result to the same storage location. Conceptually, a value of plus one is entered into the A register. The operand then reads from storage and adds to the contents of A in 18-bit ones complement logic. The operand is treated as a 16-bit positive integer in this process. An 18-bit operand is formed from the 16-bit storage operand by adding two highest order zero bits.

The result remains in A register, and bits 48 through 63 are returned to storage. The value in A register is not necessarily equal to the quantity returned to storage. The storage address for reading the operand and storing the result is formed by adding bits 52 through 63 of the m-field to bits 52 through 63 of location d in 12-bit ones complement logic. If the d-field is zero, then the storage address is given by m.

Subtract One (m+(d)) 0057 d m [SOM m,d]

Subtract one (m+(d)) reads bits 52 through 63 of an operand from storage, decreases its value by one, and returns bits 52 through 63 of the result to the same storage location. Conceptually, a value of minus one is entered into the A register. The operand then reads from storage and adds to the contents of A register in 18-bit ones complement logic. The operand is treated as a 12-bit positive integer. An 18-bit operand is formed from the 12-bit storage operand by adding six highest order zero bits. The result remains in A register, and bits 52 through 63 have four highest order zero bits added and are returned to storage. The value in A register is not necessarily equal to the quantity returned to storage. The storage address for reading the operand and storing the result is formed by adding bits 52 through 63 of the m-field to bits 52 through 63 of location d in a 12-bit ones complement logic. If the d-field is zero, then the storage address is given by m.

Subtract One (m+(d)) Long 1057 d m [SOML m,d]

Subtract one (m+(d)) long reads an operand from storage, decreases its value by one, and returns the result to the same storage location. Conceptually, a value of minus one is entered into the A register. The operand then reads from storage and adds to the contents of A register in 18-bit ones complement logic. The operand is treated as a 16-bit positive integer. An 18-bit operand is formed from the 16-bit storage operand by adding two highest order bits. The result remains in A register, and bits 48 through 63 are returned to storage. The value in A register is not necessarily equal to the quantity returned to storage. The storage address for reading the operand and storing the result is formed by adding bits 52 through 63 of the m-field to bits 52 through 63 of location d in 12-bit ones complement logic. If the d-field is zero, the storage address is given by m.

BRANCH

The branch instructions provide the capability to depart from the sequential execution of instructions.

Unconditional Jump d 0003 d [UJN d]

Unconditional jump d causes a branch to a location forward or backward as specified by d. The d-field is considered a 6-bit ones complement number. If d is in the range 0 through 31 (decimal), the branch is forward d locations. If d is in the range 32 through 63, the branch is backward 63-d locations.

Zero Jump d 0004 d [ZJN d]

Zero jump d causes a branch to a location forward or backward as specified by d when A register is zero (all bits of A register are clear). The d-field is considered a 6-bit ones complement number. If d is in the range 0 through 31 (decimal), the branch is forward d locations. If d is in the range 32 through 63, the branch is backward 63-d locations.

Non-Zero Jump d 0005 d [NJN d]

Non-zero jump d causes a branch to a location forward or backward as specified by d when A register is non-zero (all bits of A register are not clear). The d-field is considered a 6-bit ones complement number. If d is in the range of 0 through 31 (decimal), the branch is forward d locations. If d is in the range 32 through 63, the branch is backward 63-d locations.

Plus Jump d 0006 d [PJN d]

Plus jump d causes a branch to a location forward or backward as specified by d when A register is positive (bit 46 of A register is clear). The d-field is considered a 6-bit ones complement number. If d is in the range 0 through 31 (decimal), the branch is forward d locations. If d is in the range 32 through 63, the branch is backward 63-d locations.

Minus Jump d 0007 d [MJN d]

Minus jump d causes a branch to a location forward or backward as specified by d when A register is negative (bit 46 of A register is set). The d-field is considered a 6-bit ones complement number. If d is in the range 0 through 31 (decimal), the branch is forward d locations. If d is in the range 32 through 63, the branch is backward 63-d locations.

Long Jump to m+(d) 0001 d m [LJM m,d]

Long jump to m+(d) branches to an address formed from m+(d). Bits 52 through 63 of the m field are added to bits 52 through 63 of location d in 12-bit ones complement logic. The result forms an address which is used to obtain the first word of the new instruction sequence. If the d-field is zero, then the address is given by m.

Return Jump to m+(d) 0002 d m [RJM m,d]

Return jump to m+(d) stores the current program address plus two ((P)+2) in the address formed from m+(d). The instruction then branches to m+(d)+1. To form the storage address, bits 52 through 63 of the m-field are added to bits 52 through 63 of the contents of location d in 12-bit ones complement logic. If the d-field is zero, the address is given by m.

ADU INSTRUCTIONS

The ADU instructions provide the capability to read and write CM words to and from PPM. The PPs have access to all CM storage locations. CM addressing is performed with real rather than virtual memory addresses. The CM address is formed from the contents of the R register and the A register. See section IV-2, CM Addressing, for details.

Load R Register 0024 d [LRN d]

Load R register loads the 22-bit R register from (d) and (d)+1. If d is non-zero, the lowest order 12 bits (46 through 57) of R register are loaded from bits 52 through 63 of (d)+1. The remaining highest order 10 bits (36 through 45) are loaded from bits 54 through 63 of (d). If the d field is zero, then the instruction is a pass.

Store R Register 0025 d [SRD d]

Store R register stores the contents of the 22-bit R register into bits 52 through 63 of (d) and (d)+1. If d is nonzero, the lowest order 12 bits (46 through 57) of R register are stored into (d)+1. The remaining highest order 10 bits (36 through 45) are stored into bits 54 through 63 of (d). If the d-field is zero, then the instruction is a pass.

Central Read from (A) to d 0060 d [CRD d]

Central read from (A) to d transfers bits 4 through 63 of one CM word to bits 52-63 of five consecutive PPM words. Bits 0 through 3 of the CM word are discarded and the remaining 60 bits are disassembled from the left into five 12-bit words. This unpacking is illustrated below.

The address of the CM word is specified by (R)+(A). The address of the first PPM word is specified by d. This address is written into the Q register. The Q register then increments the PPM address by one every time a 12-bit word is transferred to PPM from CM.

Central Memory Word Within ADU

0	4	1	2	4	5	6
6	8	0	2	3		
(4)	a(12/3)	b(12/3)	c(12/3)	d(12/3)	e(12/3)	

PPM Words

	4	5	6
	8	2	3
d	0(4)	a(12/1)*	
d+1	0(4)	b(12/1)	
d+2	0(4)	c(12/1)	
d+3	0(4)	d(12/1)	
d+4	0(4)	e(12/1)	

Central Read from (A) to d Long 1060 d [CRDL d]

Central read from (A) to d long transfers one CM word to bits 48 through 63 of four consecutive PPM words. The CM word is disassembled from the left into four 16-bit words. This unpacking is illustrated as follows.

Central Memory Word Within ADU

0	1	3	4	6
6	2	8	3	
a(16/4)	b(16/4)	c(16/4)	d(16/4)	

PPM Words

	4	6
	8	3
d	a(16/1)	
d+1	b(16/1)	
d+2	c(16/1)	
d+3	d(16/1)	

The address of the CM word is specified by (R)+(A). The address of the first PPM word is specified by d. This address is written into the Q register. The Q register increments the PPM address by one every time a 16-bit byte is transferred to PPM from CM.

Central Read (d) Words From (A) to m 0061 d m [CRM m,d]

Central read (d) words from (A) to m transfers bits 4 through 63 of consecutive CM words to bits 52 through 63 of consecutive PPM words. Bits 0 through 4 of each CM word are discarded and the remaining 60 bits are disassembled from the left into five 12-bit words. See 0060 instruction for illustration of unpacking above.

The address of the first PP word is specified by m. This address is written into the P register while the program address is stored in PPM location 0000. The P register increments the PPM address by one every time a 12-bit word is transferred from CM to PPM. The number of CM words is specified by (d). This value is written into the Q register. Every time a CM word is transferred to PPM, Q register is decremented by one.

The address of the first CM word is specified by (R)+(A). Upon completion, A register contains the nonrelocated portion of the CM address plus part of the last CM word transferred. If the value of A register exceeds 377777, then bit 46 of A register sets. This causes the operation to switch between direct address and relocation address modes. If the last word transferred is from a relative address of 777776, then upon completion the A register is zero and does not point to the address plus one of the last word transferred.

Central Read (d) Words From (A) to m Long 1061 d m [CRML m,d]

Central read (d) words from (A) to m long transfers consecutive CM words to consecutive PPM words. Each CM word is disassembled from the left. See the 1060 instruction for illustration of this unpacking as follows.

The address of the first PPM word is specified by m. This address is written into the P register while the program address is stored in PPM location 0000. The P register increments the PPM address by one every time a 16-bit byte is transferred from CM to PPM. The number of CM words transferred is specified by (d). This value is written into the Q register. Every time a CM word is transferred to PPM, Q register is decremented by one.

The address of the first CM word is specified by (R)+(A). Upon completion, A register contains the nonrelocated portion of the CM address plus part of the last CM word transferred. If the value of A register exceeds 377777, bit 46 of A register sets. This setting causes the operation to switch between direct address and relocation address modes. If the last word transferred is from a relative address of 777776, upon completion the A register is zero and does not point to the address plus one of the last word transferred.

Central Write From d to (A) 0062 d [CWD d]

Central write from d to (A) transfers bits 52 through 63 of five consecutive PPM words (bits 48 through 51 are ignored) to bits 4 through 63 of one CM word (bits 0 through 3 are cleared). These bytes are assembled from the left as illustrated below.

The address of the first CM word is specified by (R)+(A). The address of the first PP word is specified by d. This address is written into the Q register. The Q register increments the PPM address by one every time a 12-bit byte is transferred from PPM.

PPM Words

	4	5	6
	8	2	3
d	(4)	a(12/3)	
d+1	(4)	b(12/3)	
d+2	(4)	c(12/3)	
d+3	(4)	d(12/3)	
d+4	(4)	e(12/3)	

Central Memory Word Within ADU

0	4	1	2	4	5	6
		6	8	0	2	3
(4)	a(12/3)	b(12/3)	c(12/3)	d(12/3)	e(12/3)	

Central Write From d to (A) Long 1062 d [CWDL d]

Central write from d to (A) long transfers four consecutive PPM words to one CM word. This packing is illustrated below.

The address of the CM word is specified by (R)+(A). The address of the first PPM word is specified by d. This address is written into the Q register. The Q register increments the PPM address by one every time a 16-bit byte is transferred from PPM.

PPM Word

	4	6
	8	3
d	a(16/4)	
d+1	b(16/4)	
d+2	c(16/4)	
d+3	d(16/4)	

Central Memory Word Within ADU

	1	3	4	6
	6	2	8	3
0	a(16/4)	b(16/4)	c(16/4)	d(16/4)

Central Write (d) Words From m to (A) 0063 d m [CWM m,d]

Central write (d) words from m to (A) transfers bits 52 through 63 of consecutive PPM words to bits 4 through 63 of consecutive CM words. See the 0062 instruction for illustration of this packing.

The address of the first PPM word is specified by m. This address is written into the P register while the program address is stored in PPM location 0000. The P register increments the PPM address by one every time a 12-bit byte is transferred from PPM. The number of CM words transferred is specified by (d). This value is written into the Q register. Every time a CM word is transferred from PPM to CM Q register is decremented by one.

The address of the first CM word is specified by (R)+(A). Upon completion, A register contains the nonrelocated portion of the CM address plus part of the last CM word transferred. If the value of A register exceeds 377777, then bit 46 of A register sets. This causes the operation to switch between direct address and relocation address modes. If the last word transferred is from a relative address of 777776, then upon completion the A register is zero and does not point to the address plus one of the last word transferred.

Central Write (d) Words From m to (A) Long 1063 d m [CWML m,d]

Central write (d) words from m to (A) long transfers consecutive PPM words to consecutive CM words. Four PPM words are packed from the left into each CM word.

The address of the first PPM word is specified by m. This address is written into the P register, while the program address is stored in PPM location 000. The P register increments the PPM address by one every time a 16-bit byte is transferred from PPM. The number of CM words transferred is specified by (d). This value is written into the Q register. Every time a CM word is transferred from PPM Q register is decremented by one.

The address of the first CM word is specified by (R)+(A). Upon completion, A register contains the nonrelocated portion of the CM address plus part of the last CM word transferred. If the value of A register exceeds 377777, then bit 46 of A register sets. This causes the operation to switch between direct address and relocation address modes. If the last word transferred is from a relative address of 777776, then upon completion the A register is zero and does not point to the address plus one of the last word transferred.

Central Read and Set Lock From d to (A) 1000 d [RDSL d]

Central read and set lock from d to (A) performs a logical OR function between four consecutive PPM words and one CM word with the result replacing the CM word. The original contents of the CM word replaces the four PPM words. See the 1060 and 1062 instructions for the packing and unpacking involved. The address of the first PP word is specified by d. The address of the first CM word is specified by (R)+(A).

A serialization function is performed by the memory element before this instruction begins and after it ends. Execution of this instruction is delayed until all previous accesses to CM by the IOU are completed. No other accesses from any port are permitted to the CM word from the beginning of the read to the end of the write.

Central Read and Clear Lock From d to (A) 1001 d [RDCL d]

Central read and clear lock from d to (A) performs a logical AND function between four consecutive PPM words and one CM word with the result replacing the CM word. The original contents of the CM word replace the four PPM words. See the 1060 and 1062 instructions for the packing and unpacking involved. The address of the first PPM word is specified by d. The address of the first CM word is specified by (R)+(A).

A serialization function is performed by the memory element before this instruction begins and after it ends. Execution of this instruction is delayed until all previous accesses to CM by the IOU are completed. No other accesses from any port are permitted to the CM word from the beginning of the read to the end of the write.

INPUT/OUTPUT

Twenty-three instructions direct activity on the I/O channels. These instructions select an external device and transfer data to or from that device. The instructions also determine whether a channel or external device is available and ready to transfer data. The preparatory steps ensure that data transfers in an orderly fashion.

Each external device has a set of external function codes that the PP uses to establish modes of operation and to start and stop data transfer. The devices are also capable of detecting certain errors, which they indicate to the controlling PP.

Jump to m if Channel c Active 00640 c m [AJM m,c]

Jump to m if channel c active branches to the location specified by m if channel c is active.

Test and Set Channel c Flag 00641 c m [SCF m,40B+c]

Test and set channel c flag branches to the location specified by m if the channel c flag is set; otherwise the instruction sets the channel c flag and exits. The channel C flag may be unconditionally set by setting m to P+2.

NOTE

A conflict condition occurs when two PPs in different barrels are trying to execute a 00641 instruction on the same channel simultaneously.

The PP in logical barrel zero sees the true status of the channel flag. The PP in logical barrel one sees the channel flag set. Hence, the PP in logical barrel one cannot set the channel flag when there is a conflict. This condition is true for any channel.

Jump to m if Channel c Flag Set 1064X c m [FSJM m,c]

Jump to m if channel c flag set branches to the location specified by m if the flag for channel c is set.

Jump to m if Channel c Inactive 00650 c m [IJM m,c]

Jump to m if channel c inactive branches to the location specified by m if channel c is inactive.

Clear Channel c Flag 00651 c m [CCF 40B+c]

Clear channel c flag clears the flag for channel c. The m field is required but not used.

Jump to m if Channel c Flag Clear 1065X c m [FCJM m,c]

Jump to m if channel c flag clear branches to the location specified by m if the flag for channel c is clear.

Jump to m if Channel c Full 00660 c m [FJM m,c]

Jump to m if channel c full branches to the location specified by m if channel c is full.

Test and Clear Channel c Error Flag When Set 00661 c m [SFM m,40B+c]

Test and clear channel c error flag when set branches to the location specified by m if the channel c error flag is set and clears the channel c error flag.

Jump to m if Channel c Empty 00670 c m [EJM m,c]

Jump to m if channel c empty branches to the location specified by m if channel c is empty.

Test and Clear Channel c Error Flag When Clear 00671 c m [CFM m,40B+c]

Test and clear channel c error flag when clear branches to the location specified by m when the channel c error flag is clear; otherwise, it clears the error flag.

Input to A From Channel c When Active and Full 00700 c [IAN c]

Input to A register from channel c when active and full transfers a 16-bit word from channel c to bits 48 through 63 of A register. Bits 46 and 47 of A register are cleared. The instruction waits for channel c to become active and full before executing.

NOTE

If a 12-bit external interface is used on channel c, bits 46 through 51 of A register are zeros.

Input to A From Channel c if Active and Full 00701 c [IAN 40B+c]

Input to A register from channel c if active and full transfers a word from channel c to bits 48 through 63 of A register. Bits 46 and 47 of A register are cleared. If the channel c is inactive or becomes inactive before becoming full, then no transfer takes place and the instruction exits with (A)=0.

NOTE

If a 12-bit external interface is used on channel c, bits 46 through 51 of A register are zeros.

Input (A) Words to m From Channel c 0071X c m [IAM m,c]

Input (A) words to m from channel c transfers successive words from channel c to consecutive PPM words. The address of the first PPM word is specified by m. This address is written into the P register while the program address is stored in PPM location 0000. The P register increments the PPM address by one every time a channel word is transferred into PPM. The number of words transferred is specified by (A). Every time a channel word is transferred, (A) decrements by one.

The transfer is complete when (A)=0 or channel c becomes inactive. If the termination is caused by an inactive channel, the next PPM word is cleared and A register contains the difference between its initial value and the number of words actually transferred from channel c.

If the instruction is executed with channel c initially inactive, no transfer takes place and the instruction exits with A register unchanged. The PPM word specified by m is set to 0.

NOTE

If a 12-bit external interface is used on channel c,
bits 48 through 51 of PPM are zeros.

Input (A) Words to m From Channel c Packed 1071X c m [IAPM m,c]

Input (A) words to m from channel c packed transfers bits 52 through 63 of successive words from channel c to consecutive PPM words. During this transfer, four channel words (48 bits) are packed into three PPM words illustrated below. Bits 48 through 51 of the 16-bit channel words are ignored. The address of the first PPM word is specified by m. This address is written into the P register, while the program address is stored in PPM location 0000. The P register increments the PPM address by one every time a channel word is transferred into PPM. The number of channel words to be transferred is specified by (A). Every time a channel word is transferred to PPM (A) decrements by one.

The transfer is complete when (A)=0 or channel c becomes inactive. If (A)=0 and the number of channel words transferred is not a multiple of four, the last PPM word is zero-filled only if the channel is empty at the time the last word is written into PPM (one major cycle after the last word is read from channel c). If channel c is full, then the data on channel c fills the last word being written into PPM. This zero filling of a word is dependent on channel c being empty or inactive. If the termination is caused by an inactive channel c, then PPM words are zero-filled to the next four channel c word boundary. For example, if 17 channel words were transmitted followed by an inactive, then the first 16 channel words are the first 12 PPM words (starting address m). The thirteenth PPM word contains bits 48 through 59 of the seventeenth channel word. Bits 60 through 63 with the next two PPM words are zeros.

If the instruction is executed with channel c initially inactive, no transfer takes place and the instruction exits with A register unchanged, and the next three PPM words specified by m, m+1, and m+2 are zeros.

Channel Words

4	5	6
8	2	3
(4)	a(12)	
(4)	b(4)	c(8)
(4)	d(8)	e(4)
(4)	f(12)	

PPM Words

	4	5	5	6	6
	8	2	6	0	3
m	a(12)			b(4)	
m+1	c(8)		d(8)		
m+2	e(4)	f(12)			

Output From A to Channel c 00720 c [OAN c]

Output from A to channel c transfers bits 48 through 63 of A register to channel c. The instruction waits for channel c to become active and empty before executing. The contents of A register is not altered.

NOTE

If a 12-bit external interface is used on channel c, bits 48 through 51 of the channel word are not transmitted to the external device and are lost.

Output From A to Channel c 00721 c [OAN 40B+c]

Output from A to channel c transfers bits 48 through 63 of A register to channel c. If channel c is inactive, then no transfer takes place and the instruction exits. The contents of A register are not altered.

NOTE

If a 12-bit external interface is used on channel c, then bits 48 through 51 of the channel word are not transmitted to the external device and are lost.

Output (A) Words From m to Channel c 0073X c m [OAM m,c]

Output (A) words from m to channel c transfers the contents of consecutive PPM words as successive words on channel c. The address of the first PPM word is specified by m. This address is written into the P register, while the program address is stored in PPM location 0000. The P register increments the PPM address by one every time a channel word is transferred to channel c. The number of words to be transferred is specified by A register. Every time a channel word is transferred to channel c, A register decrements by one.

The transfer is complete when either (A)=0 or the channel becomes inactive. If the termination is caused by an inactive channel c, A register contains the difference between its initial value and the number of words actually transferred on channel c. If the instruction is executed with channel c initially inactive, no transfer takes place and the instruction exits with A register unchanged.

NOTE

If a 12-bit external interface is used on channel c, then bits 48 through 51 of the channel word are not transmitted to the external device and are lost.

Output (A) Words From m to Channel c Packed 1073X c m [OAPM m,c]

Output (A) words from m to channel c packed transfers consecutive PPM words as bits 52 through 63 of successive words on channel c. During the transfer, three PPM words are unpacked into four channel c words. Bits 48 through 51 of the 16-bit channel words are cleared. This unpacking is illustrated in the 1071 instruction. The address of the first PPM word is specified by m. This address is written into the P register, while the program address is stored in PPM location 0000. The P register increments the PPM address by one every time a channel word is transferred to channel c. The number of channel words transferred is specified by A register. Every time a channel word is transferred to channel c, A register decrements by one.

The transfer is complete when (A)=0 or channel c becomes inactive. If termination is caused by an inactive channel c, A register contains the difference between its initial value and the number of words actually transferred on the channel. If the instruction is executed with channel c initially inactive, no transfer occurs and the instruction exits with A register unchanged.

Activate Channel c When Inactive 00740 c [ACN c]

Activate channel c when inactive prepares channel c for I/O transfer operations by setting channel c active. If channel c is initially active, then the instruction waits for channel c to become inactive before executing.

Unconditionally Activate Channel c 00741 c [ACN 40B+c]

Unconditionally activate channel c prepares channel c for I/O transfer operations by setting channel c active. The instruction executes regardless of the active/inactive status of channel c. The Active-Out pulse is sent to the external device only if channel c is inactive at the time of execution.

Deactivate Channel c When Active 00750 c [DCN c]

Deactivate channel c when active terminates I/O operations on channel c by setting channel c inactive. If channel c is initially inactive, then the instruction waits for channel c to become active before executing. If the instruction is executed after an output instruction without waiting for channel c to empty, then the last channel word transferred may be lost.

Unconditionally Deactivate Channel c 00751 c [DCN 40B+c]

Unconditionally deactivate channel c terminates I/O operations on channel c by setting channel c inactive. The instruction executes regardless of the active/inactive status of channel c. The Inactive-Out pulse is sent to the external device only if channel c is active at the time of execution. If this instruction is executed after an output instruction without waiting for channel c to empty, the last channel word transferred may be lost.

Function (A) on Channel c When Inactive 00760 c [FAN c]

Function (A) on channel c when inactive causes a Function signal to be transferred to the external device, while the internal channel c is set active and full. The instruction transfers bits 48 through 63 of A register to channel c as a function code. If channel c is initially active, the instruction waits for channel c to become inactive before executing. The contents of A register are not altered.

NOTE

If a 12-bit external interface is used on channel c, then bits 48 through 51 of A register are not transmitted to the external device and are lost.

Function (A) on Channel c if Inactive 00761 c [FAN 40B+c]

Function (A) on channel c if inactive causes a Function signal to be transferred to the external device, while the internal channel c is set active and full. The instruction transfers bits 48 through 63 of A register to channel c as a function code. If channel c is initially active, then the function is not transferred on channel c, and the status of channel c is unchanged. The contents of A register are not altered.

NOTE

If a 12-bit external interface is used on channel c, then bits 48 through 51 of A register are not transmitted to the external device and are lost.

Function m on Channel c When Inactive 00770 c m [FNC m,c]

Function m on channel c when inactive causes a Function signal to be transferred to the external device while the internal channel c is set active and full. The instruction transfers m to channel c as a function code. If channel c is initially active, then the instruction waits for channel c to become inactive before executing.

NOTE

If a 12-bit external interface is used on channel c, then bits 48 through 51 of the function word m are not transmitted to the external device and are lost.

Function m on Channel c if Inactive 00771 c m [FNC m,40B+c]

Function m on channel c if inactive causes a Function signal to be transferred to the external device, while the internal channel c is set active and full. The instruction transfers m to channel c as a function code. If channel c is initially active, then the function is not transferred on channel c, and the status of channel c is unchanged.

NOTE

If a 12-bit external interface is used on channel c, then bits 48 through 51 of the function word m are not transmitted to the external device and are lost.

OTHER INSTRUCTIONS

Pass 0000 d [PSN] 0024 0 0025 0 0027 x

The pass instructions perform no operation.

PP Keypoint 0027 [KEYP]

PP keypoint executes as a pass instruction but allows sensing of its execution by external monitoring equipment through a test point.

Exchange Jump 0026 d

Exchange jump provides the capability for PP programs to control the execution of the CPU in CYBER 170 mode. The Exchange Request signal is transmitted to the designated CPU for performing CYBER 170 mode execution. If that CPU is executing in CYBER 180 mode, it sets bit 53 of the monitor condition register to indicate that an IOU exchange request is outstanding. The CYBER 180 monitor must execute a CYBER 180 exchange operation to CYBER 170 mode before the IOU exchange request can be satisfied.

The instruction does not exit until an Exchange Accept signal is received from the CPU. The Exchange Accept is sent upon completion of the requested CYBER 170 exchange jump. The value of d controls the exchange request in CYBER 170 mode.

d = 0 - Unconditional Exchange Jump 0026 00 [EXN]

An exchange jump occurs unconditionally at the address specified by (R)+(A). Only the FWA of the exchange package specified by (R)+(A) is checked against the operating system (OS) boundary address.

d = 10 - Monitor Exchange Jump 0026 10 [MXN]

An exchange jump occurs conditionally at the address specified by (R)+(A). If the CPU CYBER 170 monitor flag is clear, the exchange jump occurs and sets the CYBER 170 monitor flag. If the CYBER 170 monitor flag is set, no exchange jump occurs. Only the FWA of the exchange package, as specified by (R)+(A), can be checked against the OS boundary address.

d = 20B - Monitor Exchange Jump to MA 0026 20 [MAN]

An exchange jump occurs at the address specified by the CPU monitor address register. If the CPU CYBER 170 monitor flag is clear, the exchange jump occurs and sets the CYBER 170 monitor flag. If the CYBER 170 monitor flag is set, no exchange jump occurs. If d = 30, the instruction executes as if d = 20.

If an exchange request for any PP is outstanding, a further exchange request from any PP causes that PP to wait until the previous exchange request is honoured.

Interrupt Processor 1026 d [INPN d]

Interrupt processor transmits an Interrupt signal for the CPU on the CM port specified by d. The Interrupt signal is transmitted via the CM port interface. The Interrupt signal causes the external interrupt bit to set in the CPU monitor condition register.

APPENDIX C

IOU MICRAND FIELDS

=====

All the micrand fields used by the IOU are listed in this appendix. The micrand bits originate from the branch 1 and 2 ROM (BAS 2.7) and the microcode ROM (BAS 2.8). Table C-1 (BAS 2.13) shows the IOU micrand fields.

C-2

60469460 B

C1138

NOTE 1 : (INACT 058)+(CHAN FLAG1058)

TABLE C-1. IOU MICRAND FIELDS (2 of 2)

MICRAND BIT NO.	44-48	49-58	59-68	61-70	71	72	73-75	76	77	78	79-83	84	85	86	87
SYMBOL	-BC	-BR1	-YC	-BR2	+YA	+RLY	-RVCH	-R	-W	+FF	-MCH	-CS	+FLAG	-P	+FCC
FUNCTION	BRANCH COND	BRANCH ADRS 1	Y MIX SEL	BRANCH ADRS 2	NOT CHAN DATA TO A	R UPPER SEL	ADU/CHAN CONT	ENBL READ	ENBL WRITE	FCTN OR FULL	ADU/CHAN FCTNS	CHAN SEL ENBL	PP TO PP SYNCH	PARITY	NOT FORCE CHAN CODE
PAK DESTINATION	CE 3.0	BAS 2.7	PPH 2.2	BAS 2.7	BAS 2.0	BAS 2.3	ADU 2.0 ADU 2.3 CHL 2.0 SCH 2.1 DSC 2.1	ADU 2.3	ADU 2.3	BAS 2.10	BAS 2.9 2.10 MEM 2.5	BAS 2.9	CHL 2.0 SCH 2.1 DSC 2.1	BAS 2.8	BAS 2.9

MICRAND BIT		44-48		59,60	73-75	79-83																															
SYMBOL		BC		YC	RVCH	MCH																															
CODE		A	B																																		
00		ZERO	CH BUSY + ADU BUSY	A BYPASS RK 0 DATA	BIT 76 SET (ENBL RD) <table><tr><td>0</td><td>SEL WORD 0</td></tr><tr><td>1</td><td>SEL WORD 1</td></tr><tr><td>2</td><td>SEL WORD 2 & CLEAR DATA RDY</td></tr><tr><td>3</td><td>SEL WORD 3</td></tr><tr><td>4</td><td>SEL WORD 4</td></tr><tr><td>5</td><td>SEL WORD 4, READ REQ</td></tr><tr><td>6</td><td>READ REQ IF CH NOT BUSY</td></tr><tr><td>7</td><td>NU</td></tr></table>	0	SEL WORD 0	1	SEL WORD 1	2	SEL WORD 2 & CLEAR DATA RDY	3	SEL WORD 3	4	SEL WORD 4	5	SEL WORD 4, READ REQ	6	READ REQ IF CH NOT BUSY	7	NU	10 READ (21) 12 WRITE (24) 14 READ AND SET LOCK (30) 15 READ AND CLEAR LOCK (33) 00 EXCH REQ (0) 04 INTERRUPT (11)															
0	SEL WORD 0																																				
1	SEL WORD 1																																				
2	SEL WORD 2 & CLEAR DATA RDY																																				
3	SEL WORD 3																																				
4	SEL WORD 4																																				
5	SEL WORD 4, READ REQ																																				
6	READ REQ IF CH NOT BUSY																																				
7	NU																																				
01	ZERO	ADU BUSY + DATA RDY	MAC/RA MIX DATA																																		
02	ZERO	ADU BUSY + CH BUSY	CHAN DATA																																		
03	ZERO	(FULL • A NE 1)+(ACT 1•EMP)	P RSTR RK 0 DATA																																		
04	EXCH BSY • (CH+ADU BSY)	CH BSY + ADU BSY																																			
05	ZERO	FULL + (ACT • EMP 1•A NE 1)																																			
06	A EQL 0	ACT • EMP • A NE 0																																			
07	FULL • A NE 0 1•A NE 1	ACT • EMP • A NE 0																																			
10	Q NE 0 • PP ERROR	ZERO		BIT 77 SET (ENBL WRITE) <table><tr><td>0</td><td>SEL WORD 0</td></tr><tr><td>1</td><td>SEL WORD 1</td></tr><tr><td>2</td><td>SEL WORD 2</td></tr><tr><td>3</td><td>SEL WORD 3</td></tr><tr><td>4</td><td>SEL WORD 4, WRITE REQ</td></tr><tr><td>5</td><td>SEL WORD 4, LAST WRITE REQ</td></tr><tr><td>6</td><td>EXCHANGE</td></tr><tr><td>7</td><td>NU</td></tr></table>	0	SEL WORD 0	1	SEL WORD 1	2	SEL WORD 2	3	SEL WORD 3	4	SEL WORD 4, WRITE REQ	5	SEL WORD 4, LAST WRITE REQ	6	EXCHANGE	7	NU	BIT 82 • BIT 84 1•CHAN FULL EDL FD TO C <table><tr><td>0</td><td>16 TO 12 WORD 2</td></tr><tr><td>1</td><td>16 TO 12 WORD 1</td></tr><tr><td>2</td><td>16 TO 12 WORD 4</td></tr><tr><td>3</td><td>16 TO 12 WORD 3</td></tr><tr><td>4</td><td>NU</td></tr><tr><td>5</td><td>NU</td></tr><tr><td>6</td><td>NU</td></tr><tr><td>7</td><td>16 TO 16</td></tr></table>	0	16 TO 12 WORD 2	1	16 TO 12 WORD 1	2	16 TO 12 WORD 4	3	16 TO 12 WORD 3	4	NU	5	NU	6	NU	7	16 TO 16
0	SEL WORD 0																																				
1	SEL WORD 1																																				
2	SEL WORD 2																																				
3	SEL WORD 3																																				
4	SEL WORD 4, WRITE REQ																																				
5	SEL WORD 4, LAST WRITE REQ																																				
6	EXCHANGE																																				
7	NU																																				
0	16 TO 12 WORD 2																																				
1	16 TO 12 WORD 1																																				
2	16 TO 12 WORD 4																																				
3	16 TO 12 WORD 3																																				
4	NU																																				
5	NU																																				
6	NU																																				
7	16 TO 16																																				
11	INACT + A EQL 1 •ACT 1•EMP	FULL		BIT 84 SET (FCTN OR FULL) <table><tr><td>0</td><td>FUNCTION</td></tr><tr><td>1</td><td>ACTIVE</td></tr><tr><td>2</td><td>FULL</td></tr><tr><td>3</td><td>EMPTY</td></tr><tr><td>4</td><td>INACTIVE</td></tr><tr><td>5</td><td>CLEAR ERROR</td></tr><tr><td>6</td><td>CLEAR FLAG</td></tr><tr><td>7</td><td>DO NOTHING</td></tr></table>	0	FUNCTION	1	ACTIVE	2	FULL	3	EMPTY	4	INACTIVE	5	CLEAR ERROR	6	CLEAR FLAG	7	DO NOTHING	(BIT 83 • BIT 84 EDL CHAN TO C) <table><tr><td>0</td><td>12 TO 16 WORD 1</td></tr><tr><td>1</td><td>12 TO 16 WORD 2</td></tr><tr><td>2</td><td>12 TO 16 WORD 3</td></tr><tr><td>3</td><td>NU</td></tr><tr><td>4</td><td>NU</td></tr><tr><td>5</td><td>NU</td></tr><tr><td>6</td><td>NU</td></tr><tr><td>7</td><td>16 TO 16</td></tr></table>	0	12 TO 16 WORD 1	1	12 TO 16 WORD 2	2	12 TO 16 WORD 3	3	NU	4	NU	5	NU	6	NU	7	16 TO 16
0	FUNCTION																																				
1	ACTIVE																																				
2	FULL																																				
3	EMPTY																																				
4	INACTIVE																																				
5	CLEAR ERROR																																				
6	CLEAR FLAG																																				
7	DO NOTHING																																				
0	12 TO 16 WORD 1																																				
1	12 TO 16 WORD 2																																				
2	12 TO 16 WORD 3																																				
3	NU																																				
4	NU																																				
5	NU																																				
6	NU																																				
7	16 TO 16																																				
12	ACT • EMP 1•A NE 1	FULL		76,77 = 1 <table><tr><td>1</td><td>CLR EXCH RESP & BUSY</td></tr></table>	1	CLR EXCH RESP & BUSY	CU38A																														
1	CLR EXCH RESP & BUSY																																				
13	A EQL 0	ZERO																																			
14	Q LT 3	ZERO																																			
15	Q EQL 0+PP ERR+CH BUSY+ADU BUSY	CH BUSY + ADU BUSY																																			
16	ONE	ONE																																			
17	ONE	ONE																																			
20	ZERO	ADU BUSY																																			
21	ZERO	ONE																																			
22	ZERO	EXCH BUSY + CH BUST + ADU BUSY																																			
23	ZERO	ZERO																																			

A	B	
0	0	BRANCH 1
0	1	NO BRANCH
1	0	BRANCH 2
1	1	776 LOAD MODE
1	1	775 DUMP MODE

MICRAND BIT ASSIGNMENT

<u>BIT NO</u>	<u>SYMBOL</u>	<u>FUNCTION</u>
00	LRU	LOAD R-UPPER
01	LRL	LOAD R-LOWER
02	PQ	P TO Q
03	SHF	SHIFT
04	FA	F TO A
05	GEA	GE TO A
06	DA	D TO A
07	QA	Q TO A
08-12	ADA	A - ADDER CODE BITS 0-4
13-14	ADQ	Q - ADDER CODE BITS 0, 1 (Q SUBTRACT, LOAD Q)
15	FQ	F TO Q
16	DQ	D TO Q
17	PDEC	P-ADDER CODE (DECREMENT P)
18	FDP	FD TO P
19	QP	Q TO P
20	ZG	ZERO TO G
21	ADUY	ADU TO Y
	ACH	A TO CHANNEL
22	16 BIT	ENABLE 16-BIT MODE
23	PASS	EXIT (24+25 INSTRUCTION)
24-26	AC	A SELECT CONDITION
27-29	QC	Q SELECT CONDITION
30-33	PC	P SELECT CONDITION
34-37	GC	G SELECT CONDITION
38-40	EC	EXIT CONDITION
41-43	WC	PP WRITE CONDITION
44-48	BC	BRANCH CONDITION
49-58	BR1	BRANCH ADDRESS 1
59-60	YC	Y MUX SELECT
61-70	BR2	BRANCH ADDRESS 2
71	YA	CHANNEL DATA TO A
72	RUY	R-UPPER SELECT
73-75	RWCH	ADU/CHANNEL CONTROLS (WORD COUNT/CHANNEL CODE)
76	R	ENABLE READ
77	W	ENABLE WRITE
78	FF	FUNCTION OR FULL
79-83	MCH	ADU/CHANNEL FUNCTIONS
84	CS	CHANNEL SELECT ENABLE
85	FLAG	PP TO PP SYNCH
86	P	PARITY (EXCLUDE BR1 & BR2)
87	FCS	FORCE CHANNEL SELECT

A ADDER CODE BITS 0-4 (MICRAND BITS 8 through 12)

<u>CODE</u>	<u>FUNCTION</u>
00	A INPUT TO A
01-03	NOT USED
04	A AND B
05	B INPUT TO A
06-07	NOT USED
10	A AND NOT B
11	A EXCLUSIVE OR B
12	B INPUT INVERT
13-25	NOT USED
26	A MINUS B
27-30	NOT USED
31	A PLUS B
32-37	NOT USED

Q ADDER CODE BITS 0, 1 (MICRAND BITS 13 and 14)

<u>CODE</u>	<u>FUNCTION</u>
00	A PLUS B
01	B INPUT TO Q
02	A MINUS B
03	KEYPOINT

A SELECT CONDITION (MICRAND BITS 24 through 26)

<u>CODE</u>	<u>FUNCTION</u>
00	FULL OR INACTIVE*
01	EMPTY*
02	ADU BUSY
03	CM BUSY
04	CM BUSY AND OUTSTANDING REQUEST
05	ADU BUSY OR CM BUSY
06	ZERO
07	ONE

Q SELECT CONDITION (MICRAND BITS 27 through 29)

<u>CODE</u>	<u>FUNCTION</u>
00	ADU BUSY OR NOT DATA READY
01	ADU BUSY
02	(Q = 0) OR PP ERROR
03	ZERO
04	ONE
05	NOT USED
06	ONE
07	NOT USED

P SELECT CONDITION (MICRAND BITS 30 through 33)

<u>CODE</u>	<u>FUNCTION</u>
00	ONE
01	ONE
02	ONE
03	ONE
04	ADU BUSY OR NOT DATA READY
05	ADU BUSY
06	ADU BUSY OR CM BUSY
07	ONE
10	FULL OR INACTIVE*
11	ACTIVE AND EMPTY*
12	ACTIVE AND EMPTY AND A≠0*
13	NOT EXCHANGE RESPONSE
14	ACTIVE AND NOT Q58*
15	NOT USED
16	ZERO
17	ONE

G SELECT CONDITION (MICRAND BITS 34 through 37)

<u>CODE</u>	<u>FUNCTION</u>
00	A GREATER OR EQUAL TO ZERO
01	A LESS THAN ZERO
02	A EQUAL TO ZERO
03	A NOT EQUAL TO ZERO
04	d NOT EQUAL TO ZERO
05	ONE
06	ADU BUSY OR CM BUSY
07	NOT USED
10	NOT (EMPTY AND NOT Q58 OR NOT CHAN ERROR AND Q58)*
11	EMPTY AND NOT Q58 OR NOT CHAN ERROR AND Q58*
12	NOT (INACTIVE AND NOT Q58 OR NOT CHAN FLAG AND Q58)*
13	INACTIVE AND NOT Q58*
14	CHANNEL FLAG
15	NOT CHANNEL FLAG
16	ONE
17	ZERO

EXIT CONDITION (MICRAND BITS 38 through 40)

<u>CODE</u>	<u>FUNCTION</u>
00	FULL OR INACTIVE AND Q58*
01	ACTIVE AND EMPTY OR INACTIVE AND Q58*
02	ACTIVE OR Q58*
03	INACTIVE OR Q58*
04	EXCHANGE RESPONSE
05	NOT (CM BUSY OR ADU BUSY)
06	D=0 OR EXIT
07	ZERO

PP WRITE CONDITION (MICRAND BITS 41 through 43)

<u>CODE</u>	<u>FUNCTION</u>
00	FULL OR INACTIVE*
01	ADU NOT BUSY
02	ADU NOT BUSY OR CM NOT BUSY
03	ADU NOT BUSY AND DATA READY
04	FULL OR INACTIVE OR A = 0*
05	LONG DEADSTART
06	ONE
07	ZERO

BRANCH CONDITION (MICRAND BITS 44 through 48)

<u>SEL</u>	<u>NO BRANCH</u>	<u>BRANCH 2</u>
00	ZERO	CM BUSY
01	ZERO	ADU BUSY OR DATA NOT READY
02	ZERO	ADU BUSY OR CM BUSY
03	ZERO	FULL AND A≠1 OR ACTIVE AND EMPTY*
04	NOT USED	NOT USED
05	ZERO	FULL OR ACTIVE AND EMPTY AND A≠1*
06	A = 0	ACTIVE AND EMPTY AND A≠0*
07	FULL AND A≠0 AND A≠1*	ACTIVE AND EMPTY AND A≠0*
10	NOT Q=0 AND NOT PP ERROR	ZERO
11	INACTIVE OR A = 1 AND ACTIVE AND EMPTY*	FULL
12	ACTIVE AND EMPTY AND A≠1*	FULL
13	A = 0	ZERO
14	Q LESS THAN 3	ZERO
15	NOT (Q = 0 OR PP ERROR OR CM BUSY OR ADU BUSY)	CM BUSY OR ADU BUSY
16	ONE	ONE
17	ONE	ONE
20	ZERO	ADU BUSY
21	ZERO	ONE
22	ZERO	EXCHANGE BUSY OR CM BUSY
23	ZERO	ZERO

Y MUX SELECT - BRANCH 1 ON BAS 2.7 (MICRAND BITS 59 and 60)

<u>CODE</u>	<u>FUNCTION</u>
00	A BYPASS RANK 0 DATA TO Y MUX
01	MAC/RA MUX DATA TO Y MUX
02	CHANNEL DATA TO Y MUX
03	P REGISTER RANK 0 DATA TO Y MUX

ADU/CHANNEL CONTROLS (WORD COUNT/CHANNEL CODE)

MICRAND BITS 73 through 75

BIT 76 SET (ENABLE READ)

<u>CODE</u>	<u>FUNCTION</u>
00	SELECT BYTE 0
01	SELECT BYTE 1
02	SELECT BYTE 2 AND CLEAR DATA READY
03	SELECT BYTE 3
04	SELECT BYTE 4
05	SELECT BYTE 4, READ REQUEST
06	READ REQUEST IF CM NOT BUSY
07	NOT USED

BIT 77 SET (ENABLE WRITE)

<u>CODE</u>	<u>FUNCTION</u>
00	SELECT BYTE 0
01	SELECT BYTE 1
02	SELECT BYTE 2
03	SELECT BYTE 3
04	SELECT BYTE 4, WRITE REQUEST
05	SELECT BYTE 4, LAST WRITE REQUEST
06	EXCHANGE
07	NOT USED

BIT 78 SET (FUNCTION OR FULL)

<u>CODE</u>	<u>FUNCTION</u>
00	FUNCTION
01	ACTIVE
02	FULL
03	EMPTY
04	INACTIVE
05	CLEAR ERROR
06	CLEAR FLAG
07	DO NOTHING

ADU/CHANNEL FUNCTIONS (MICRAND BITS 79 through 83)

<u>CODE</u>	<u>FUNCTION</u>
10	READ
12	WRITE
14	READ AND SET LOCK
15	READ AND CLEAR LOCK
00	EXCHANGE REQUEST
04	INTERRUPT

BIT 82 AND CHANNEL SELECT (BIT 84) AND
CHANNEL NOT FULL EQUALS FD TO C

<u>CODE</u>	<u>FUNCTION</u>
0	16 TO 12 WORD 2
1	16 TO 12 WORD 1
2	16 TO 12 WORD 4
3	16 TO 12 WORD 3
4	NOT USED
5	NOT USED
6	NOT USED
7	16 TO 16

BIT 83 AND CHANNEL SELECT (BIT 84) AND CHANNEL FULL OR BIT
83 AND BIT 84 AND CHANNEL NOT ACTIVE EQUALS CHANNEL TO C

<u>CODE</u>	<u>FUNCTION (CHANNEL TO PP)</u>
0	12 TO 16 WORD 1
1	12 TO 16 WORD 2
2	12 TO 16 WORD 3
3	NOT USED
4	NOT USED
5	NOT USED
6	NOT USED
7	16 TO 16

* FULL = DS FULL OR CHANNEL FULL (CE 3.0)
INACTIVE = NOT (DS SEQUENCE OR CHANNEL ACTIVE) (CE 3.0)
ACTIVE = DS SEUQNCE OR CHANNEL ACTIVE (CE 3.0)
EMPTY = NOT DS FULL AND NOT CHANNEL FULL (CE 3.0)

APPENDIX D

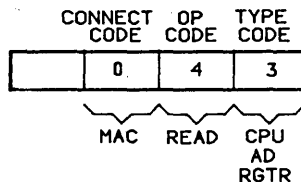
MAC/MICROCODE AIDS

AIDS FOR UNDERSTANDING MAC/MICROCODE OPERATIONS

Examples on the following pages show PP and MAC entries to read CPU registers. Table D-1 specifies the microcode starting address. Figures D-1 through D-3 show an excerpt from DQ pak, MAC internal interface, and MAC paths to/from maintenance registers, respectively.

GENERAL EXAMPLE: P.P. TO READ CPU'S P REGISTER VIA AD REGISTER

P.P. FUNCTION CODE OUTPUTS TO CHAN 17:



IN MAC, TYPE CODE=LEFT SHIFTED 1 R/W BIT

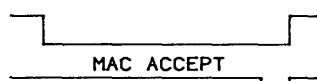
•g. 3 → 6 1=WRITE
 6= READ AD RGTR
 7= WRITE AD RGTR

MAC REQUEST TO CPU — RECOGNIZED BY MICROCODE LEADING TO:

P.P. CONTROL WORDS OUTPUT AS 2 WORDS, THE FIRST WORD IS IGNORED.

X	X	X	X	C.W. 1 (IRRELEVANT)
0	0	4	0	C.W. 2 (40 H= P REGISTER NUMBER)

EXECUTE REQUEST TO MAC



READ CONTROL WORD FROM MAC FROM THE P.P.
 PUT IT IN THE CUFFIL (RF LOCATION CA)
 DIRECT CUFFIL → COPY ROM
 THIS ADDRESSES THE MICRAND TABLE IN CPU MICROCODE
 (130=READ CPU/MAC
 131=WRITE CPU/MAC)

PICKS THE MICRAND APPLICABLE

THE CPU MICRAND NOW DIRECTS THE FOLLOWING:
 EXAMPLE: READ P RGTR:

P REGISTER → MAP SENSE MUX → IMMED MUX . BD MUX WD REGISTER → RF (CUFFIL)

* (FLAG 2=1)

* (FLAG2=0)

EXIT THE MAC REQUEST:

BUFIL=650=
 LATCH=8=LENJ

01100 1010 000
 C A

J CNTR BNR3

THIS LOOPS
 UNTIL LENJ
 IS REDUCED
 TO ZERO

RFADRS
 RF
 AD

P REGISTER CONTENT=WRITE DATA(DISASSEMBLED)

DISASSEMBLE TO BYTES

ACCEPTED BY MAC

TRANSFERRED TO CHAN 17

READ BY THE P.P.

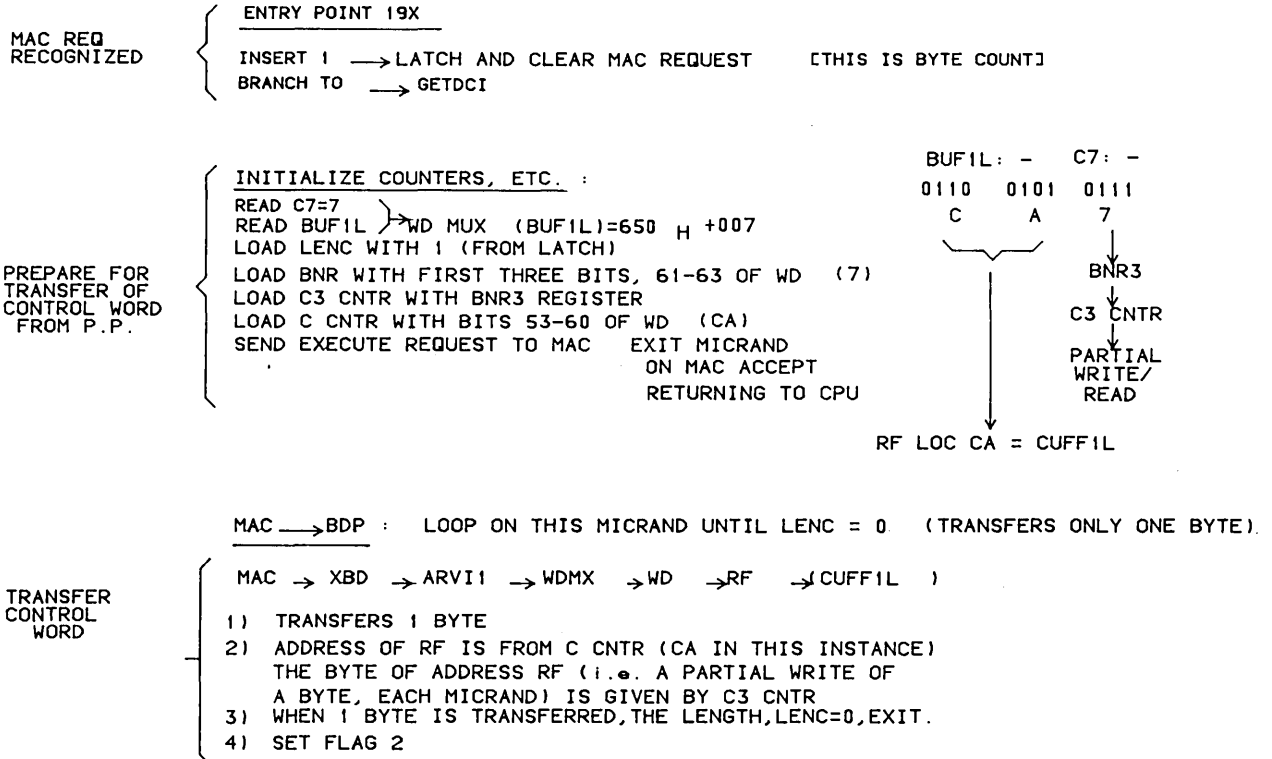
CUFFIL → XK

THIS IS THE 'COPY' INSTRUCTION
 CPU WANTS TO READ THE
 P REGISTER

C1097

NOTE: FLAG 2 WILL BE SET FOR A MAC OPERATION
 FLAG 2 WILL BE CLEAR FOR THE CPU'S COPY INSTRUCTION OPERATION
 THE SAME MICROCODE ROUTINE IS USED IN BOTH CASES FLAG 2 INDICATING WHICH APPLIES.

MAC ENTRY TO READ A CPU RESIDENT REGISTER



DUMMY : THIS IS THE END OF READING THE CONTROL WORD TO FIND OUT WHICH REGISTER IS TO BE TRANSFERRED TO THE P.P.'S, VIA MAC.

READ CUFFIL → AD → COPY ROM → S REG [COPY ROM - 8 BITS CONVERTED TO 5 BITS PROCESSOR STATE REGISTER ADDRESSING]

BRANCH TO REF 130 [COPY TO XK PER (XJ)]...130-1 TO BE MORE PRECISE.

(OE : REF 130 : COPY TO XK PER (XJ) ROUTINE USED IN MICROCODE SEQUENCE)

130-1,2 : LOAD 8 → LATCH AND CLEAR CUFFIL

AND
 130-END USE COPY ROM FROM EARLIER MICRAND TO SELECT THE APPROPRIATE READ ROUTINE FROM TABLE
 TRANSFER THE SELECTED REGISTER CONTENTS TO CUFFIL

130-MAC 1,2: TRANSFER, BYTE BY BYTE, THE REQUESTED REGISTER CONTENTS. FROM CUFFIL TO CHAN 17 AND THUS TO THE P.P., TO COMPLETE THE OPERATION.

C1098

EXAMPLE 1: READ P REGISTER: 130-140 P REGISTER NUMBER=40

(P REGISTER → MAP SENSE MUX → IMMED MUX → BD MUX → BD WD → RF LOC CUFFIL (LOC CA)
 ENTRY → 130-END: IF FLAG 2 SET CONTINUE (NOT SET ...CUFFILL → XK.)
 FROM OTHER THIS IS A NORMAL COPY INSTRUCTION
 EXAMPLES 130-MAC 1: EXIT MAC REQUEST:

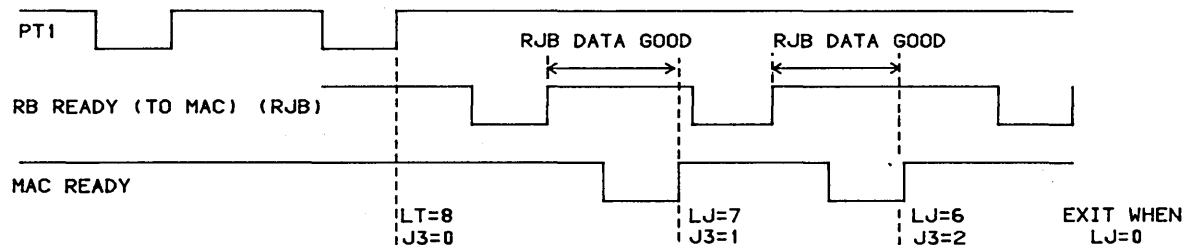
READ BUFIL 53 01100 1010 63 000
 LATCH → LENJ C A
 MAC → W CHANNEL
 ADRS OF RF LOC CA = CUFFILL
 J CNTR
 (THIS IS GOING TO STREAM BYTES TO MAC FROM CUFFILL I.E. WRITE DATA)

130-MAC 2: LOOP UNTIL ALL BYTES TRANSFERRED TO MAC TO CHAN 17
 AND THEN EXIT WHEN LENC=0

CUFFIL → AD → RJB MUX → DQ MUX → DQ REGISTER → CHANNEL REGISTER → CHANNEL 17₈

STREAMING +1 → J3 CNTR → DISASSEMBLE

-1 → LENC CNTR → WHEN =0, EXIT



C1099

EXAMPLE 2:

READ 1) STATUS SUMMARY
2) ENVIRONMENT CONTROL
3) DEPENDENT ENVIRONMENT CONTROL
4) CONTROL STORE ADDRESS
5) CONTROL STORE BREAKPOINT

(BRANCH TO 130.NP)

THESE REGISTERS USE THE CPU OR CM/IOU BIDIRECTIONAL BUSES.
THE 'COPY' ROM FOR THESE REGISTERS TO BE READ BY MAC, ADDRESSES
THE 130 TABLE AS NORMAL, BUT BRANCHES TO THE NO-OP ENTRY---WHICH
ZEROISES THE WORKING BUFFER CUFFIL. FROM THIS POINT ON, THE TRANSFER
PROCEEDS AS NORMAL i.e. AS IN THE '1ST' EXAMPLE.

EXAMPLE 3:

READ 1) MON CONDITION REGISTER (RF A5)
2) USER CONDITION REGISTER (RF A4)
3) USER MASK REGISTER (RF A2)

(BRANCH TO CUFIR30)

THESE REGISTERS, WHEN READY BY MAC, CONSIST OF THE RIGHT 48 BITS
FROM A2, A4, A5, BUT THE LEFT 16 BITS COME FROM THE HARD PSR
REGISTERS i.e. FROM THE TJ PAK.
THE CONCATENATED 64 BIT RESULT IS WRITTEN IN THE BUFFER CUFFIL.
THE BUFFER IS THEN READ AGAIN AND SHIFTED 48 DECIMAL PLACES
TO THE RIGHT, TO LEAVE THE 16 BITS i.e. THE UCR OR MCR IN THE
16. BYTE POSITIONS AND WRITTEN BACK TO CUFFIL.
THEN THE TRANSFER TO MAC OCCURS--THE SAME AS '1ST EXAMPLE'.

CPU READ MAC RESIDENT REGISTERS

EXAMPLE 4:

READ 1) ELEMENT ID
2) PROCESSOR ID
3) OPTIONS INSTALLED
4) PROCESSOR FAULT STATUS 0,1
5) RETRY CORRECTED ERROR LOG
6) CONTROL MEMORY ERROR LOG
7) CACHE CORRECTED ERROR LOG
8) MAP CORRECTED ERROR LOG
9) PROCESSOR TEST MODE

ENTER 130-0, 130-1,
130-2, 130-XX, MACCER,
130-END, 130-Xk

TO RF ONLY
i.e. Xk

(THESE REGISTERS ARE ALREADY IN MAC [OR CONTROL STORE] AND
WOULD NOT BE READ BY USING MICROCODE BY MAC i.e. FLAG 2 WOULDN'T BE SET

IF CPU WISHES TO READ THESE REGISTERS, TO THE RF; -

ENTRY WOULD BE VIA NORMAL 'COPY TO THE XK PER XJ' INSTRUCTION (REF 130,0E,
WHICH AND'S OUT THE XJ VALUE TO INDICATE WHICH OF THE ABOVE REGISTERS
REQUIRED. THEN TO '130=1 IN MICROCODE AS IN 1ST. EXAMPLE OF P REGISTER, READ
BY MAC. THIS SETS UP THE STREAMING OPERATION TO READ THE SELECTED REGISTER
ABOVE INTO CUFFIL.... 1) STREAM IS SET UP FROM COPY ROM DIRECTED MICROCODE
2) TRANSFER OF BYTES IS IN THE MACCER MICROCODE
BECAUSE FLAG 2 IS NOT SET FOR THESE REGISTERS' READING, THE 130-END MICROCODE
MOVES ON TO 130-XK MICROCODE WHICH DIRECTS CUFFIL (i.e. THE WANTED REGISTER
CONTENTS) TO XK.

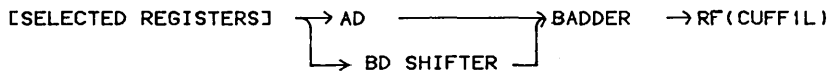
C1100

MAC ENTRY TO READ CPU REGISTERS

<u>EXAMPLE 5:</u>	<u>READ</u>		<u>RF LOCATION</u>
	0) VIRTUAL MACHINE CAPABILITY LIST	(VMCL)	3F
	1) MON PROC STATE	(J/MPS)	26
	2) UNTRANSLATABLE POINTER	(UTP)	21
	3) SEG TABLE LENGTH	(STL)	FO"NOTE"
	4) SEG TABLE ADDRESS	(STA)	EF"NOTE"
	5) PAGE TABLE ADDRESS	(PTA)	54
	6) MODEL DEPENDENT FLAGS		E
	7) MODEL DEPENDENT WORD	(TRACE)	20
	8) SYSTEM INTERVAL TIMER		
	9) TRAP POINTER		
	10) DEBUG LIST POINTER		
	11) DEBUG MASK		
	12) RF DUMP ADDRESS		

NOTE THESE VALUES ARE IN THE EXCHANGE PACKAGE AREA ALSO,BUT NOT READ FROM THESE LOCATIONS. THEY ARE READ FROM THE LOWER PART OF R.F.

THE ABOVE REGISTERS WILL BE ADDRESSED FROM THE COPY ROM INTO THE MODE, WITH FLAG 2 SET SOME OF THEM WILL BE SHIFTED UP TO 48₁₀ AS THEY ARE PROCESSED THROUGH THE SEQUENCE. -



THEN THE DISASSEMBLY BEGINS AFTER THE STREAMING COUNTERS ARE SET UP ETC,

CUFFIL → AD → RJB → MAC

MAC ENTRY TO READ CPU REGISTERS

<u>EXAMPLE 6:</u>	<u>READ:</u>		<u>RF LOCATION</u>
	1) BASE CONSTANT	(AC,AD)	C,D
	2) PAGE TABLE LENGTH		55
	3) PAGE SIZE MASK		56
	4)JOB PROCESS STATE POINTER		26
	5) TRAP ENABLES	(A1)	1
	6) KEYPOINT CODE	(A9)	9
	7) KEYPOINT CLASS NUMBER		6
	8) PROCESS INTERVAL TIMER	(AA,AB)	A,B
	9) CRITICAL FRAME FLAG	(A1)	1
	10) ON CONDITION FLAG	(A1)	1
	11) DEBUG INDEX		23

THESE REGISTERS REQUIRE VARIOUS AMOUNTS OF POSITIONING TO GET THEIR CONTENTS TO THE 1'S END OF THE BUFFER CUFFIL, BEFORE BEING STREAMED TO MAC.

C1101

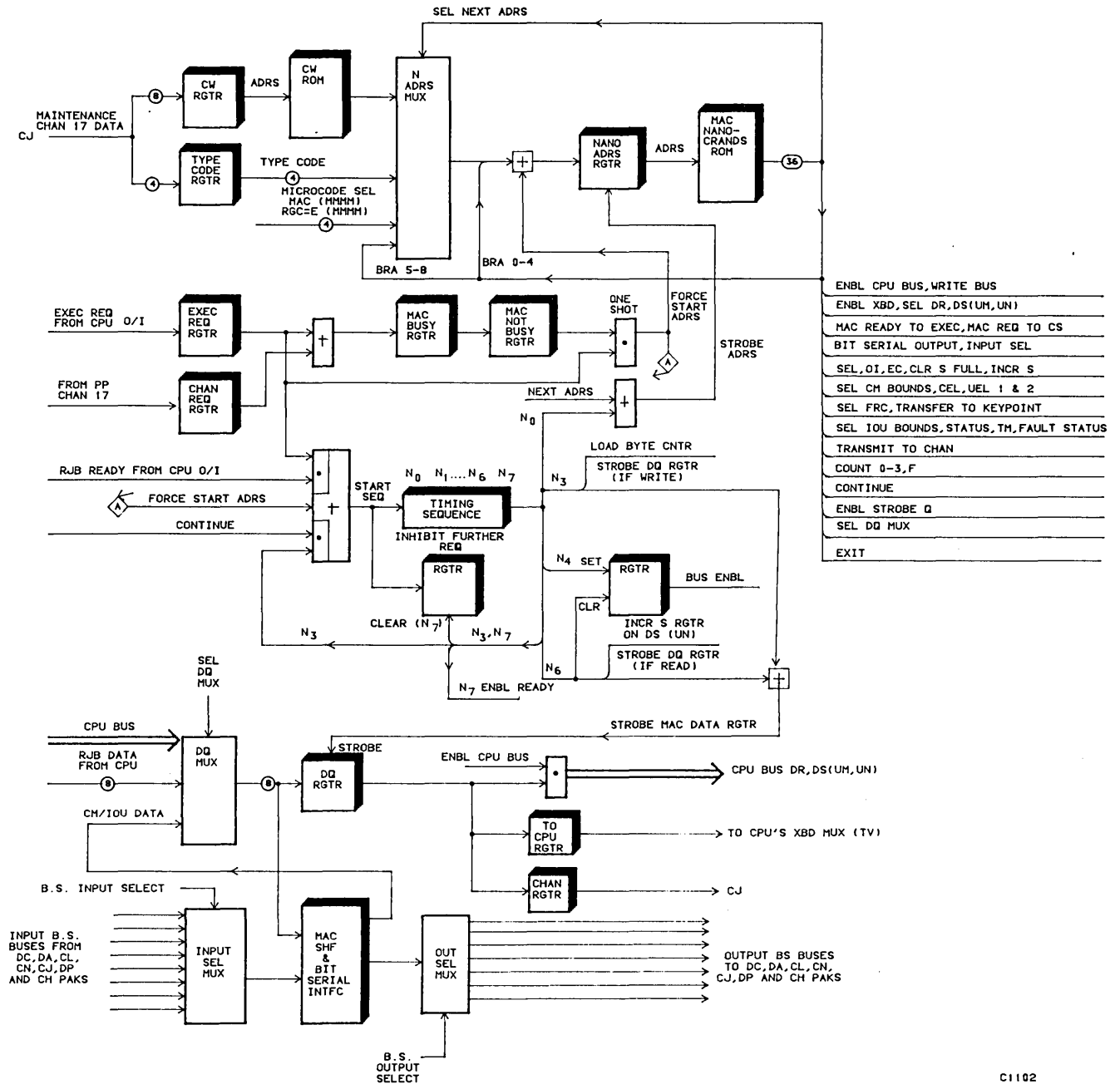
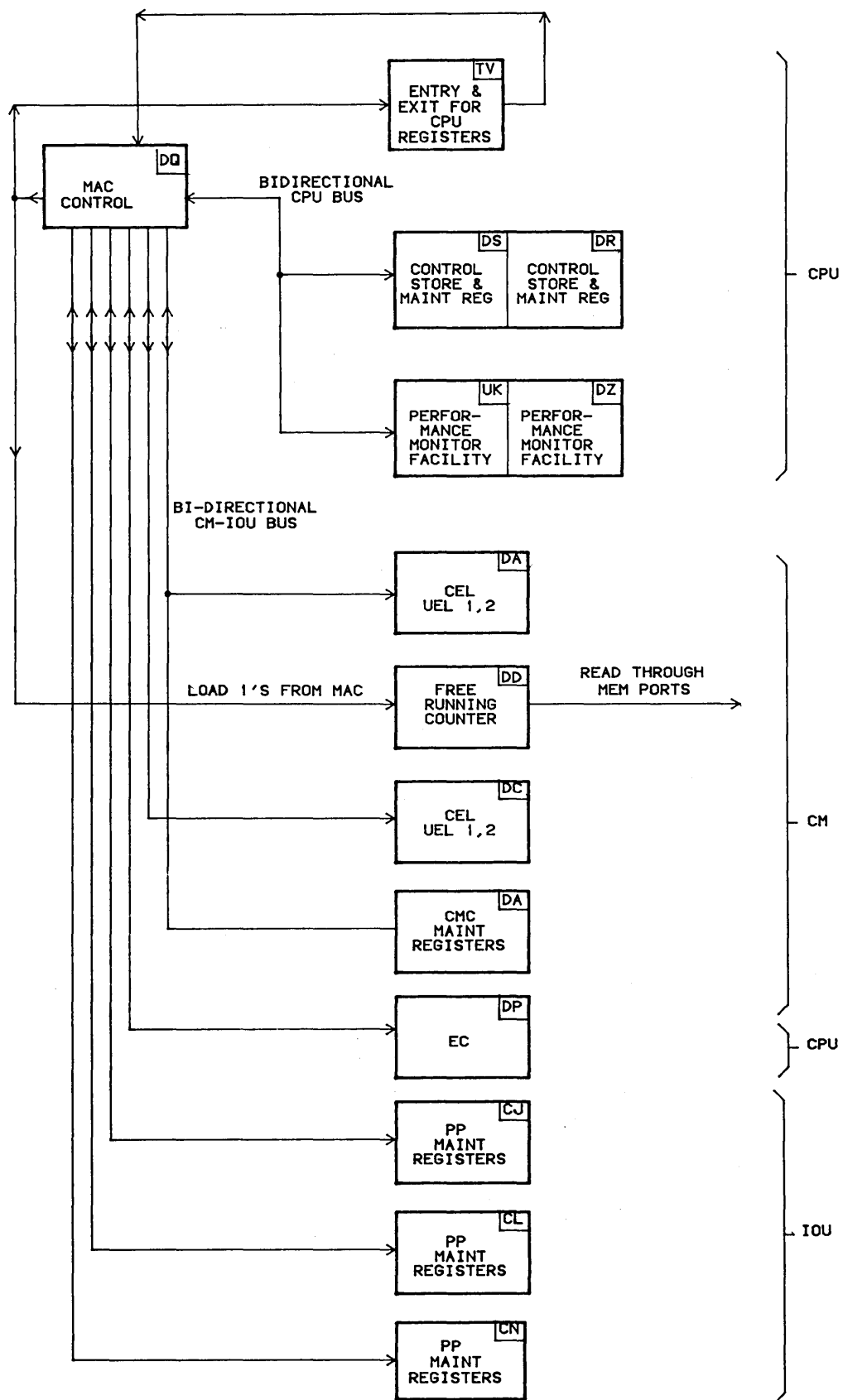


Figure D-1. Excerpt from DQ Pak



C1103

Figure D-2. MAC Internal Interface

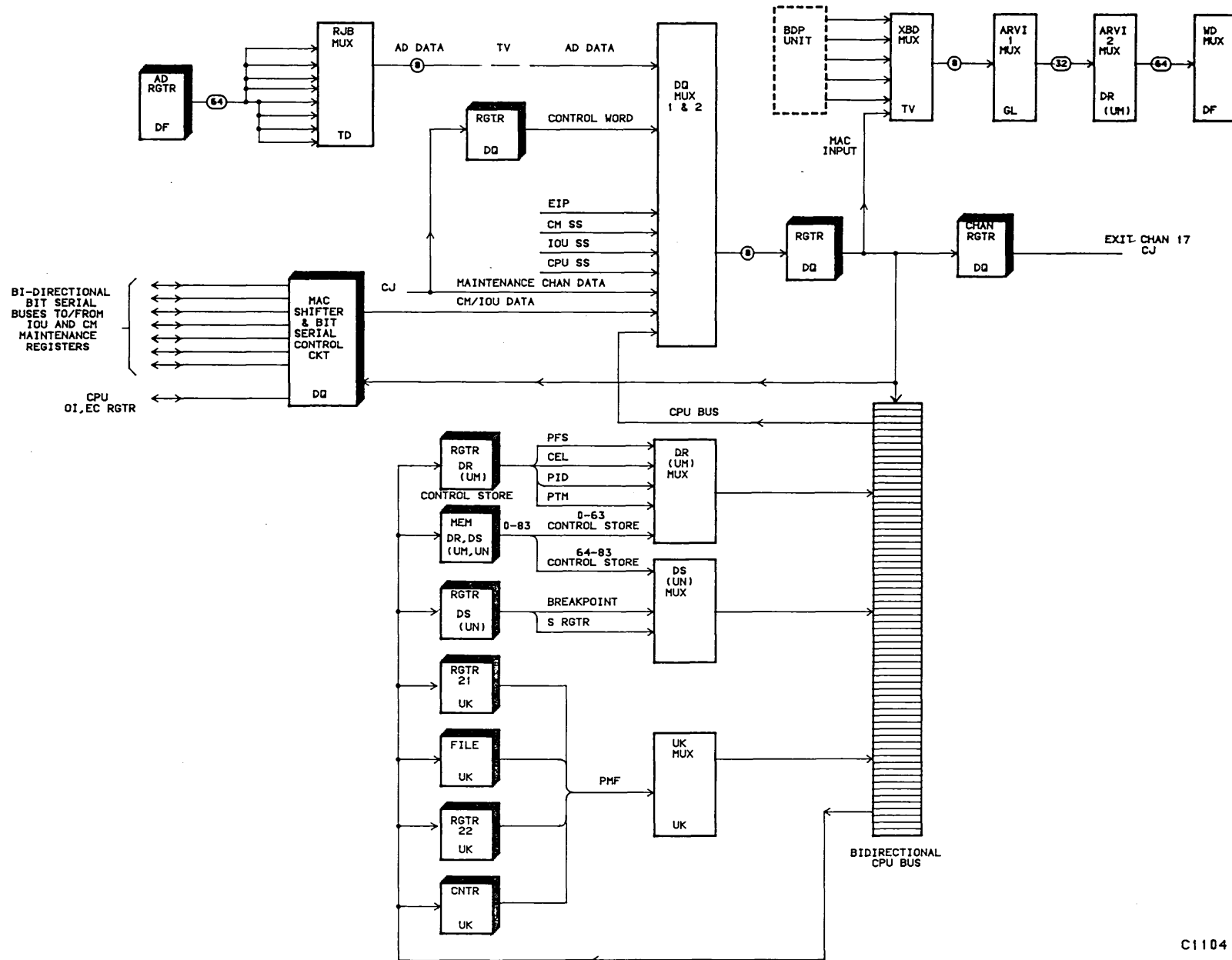


Figure D-3. MAC Paths to/from Maintenance Registers

TABLE D-1. MICROCODE STARTING ADDRESS (19X HEX)

Type Code	Read	Write	Description
F	1110	1111	Single transfer
1	0010	0011	Register file A
2	0100	0101	MAP
3	0110	0111	AD register
4	1000	1001	Maintenance scan
5	1010	1010	Control store

APPENDIX E
CPU MICRAND FIELDS

This appendix explains the micrand fields, the formats under which they are valid, and the actions they cause.

WRITE MUX SELECT (WMX)

The WMX field is a four-bit micrand field located in bits 0 through 3 of a micrand which selects the data to the WD mux on the DF pak. WMX also selects the micrand format and the unit to be active during primitive time. Unspecified bits of WD mux are set to zero.

<u>Code</u>	<u>Unit</u>	<u>Format</u>	<u>Description</u>
0000	CSU	D	WD mux selects CSU Data with appropriate ring bits; ring bit selection is controlled by micrand bits 49, 50, and 51.
0001			Not used.
0010	EXC	E	Selects 0 into WD mux bits 0 through 27, rightmost 18 bits of AD into WD mux bits 28 through 45, and rightmost 18 bits of BD into WD mux bits 46 through 63. Used for 60-bit mode exchange.
0011	COM	C	Compare result bits are selected into WD mux bits 32 and 33.
0100	BDP	G	XBD data is selected into every byte of WD mux.
0101	LAT	E	S adder latch is selected into bits 46 through 63 of WD mux. Other bits of WD mux will be ones during format E if latch bit 0 is a one and if SGNX micrand bit is one. Otherwise, other WD mux bits are zero.
0110	YKW	B	YKW[A] counter is selected left three into WD mux. All other WD mux data are zeros.
0111	QCO	B	Q counter is selected into WD mux bits 55 through 63. Other WD mux bits are zeros.
1000	PAC	A	The 16 bits of packer are selected into bits 0 through 15 of WD mux. Remainder of WD mux selects the output of L adder.
1001	LAD	B	The 15 bits of packer are selected into bits 0 through 14 of WD mux, and B adder bit 0 into bit 15 of WD mux. Remainder of WD mux selects output of L adder.
1010			Not used.
1011			ER register is selected into WD mux.
1100	AD	D	AD bits 0 through 31 with appropriate ring bits are selected into bits 0 through 31 of WD mux. The remainder of WD mux selects least significant 32 bits of B adder.

<u>Code</u>	<u>Unit</u>	<u>Format</u>	<u>Description</u>
1101	BAD	B	B adder bits 0 through 63 are selected into WD mux bits 0 through 63.
1110	PAR	H	B adder bits 0 through 31 are selected into WD mux bits 0 through 31 and 32 through 63.
1111	F	F	B adder bits 0 through 31 are selected into WD mux bits 0 through 31 and 32 through 63.

REGISTER FILE A, B, C - RGA, RGB, RGC

<u>RGA</u>	<u>Mnemonic</u>	<u>Description</u>
1XXXXXXXX	NAME	Use XXXXXXXX as an address at read time.
00000YYYY	YYYY	Use instruction fields as designated by YYYY as an address for RFA. YYYY is described in last part of this appendix.
00001YYYY	YYYY	Behaves in same way as a 00000YYYY code with exception that if selected address specified by the YYYY is the X0 register, then zeros are selected into AD mux.
00010YYYY	YYYY	Use immediate data as designated by YYYY as an address for RFA. See description of YYYY when RGA code is 0010YYYY.
00100ZZZZ	JK.48	Used during a store multiple streaming type operation to select J counter for address of A register that is currently being stored and K counter for X register being stored. Uppermost 16 bits of A registers are not stored.
00101ZZZZ	JK.64	Identical to 00100YYYY code except that A registers are stored. See process register circuit description. ZZZZ is not used.
00110ZZZZ	J.BNR	Used when J counter is used for RF address of the word to be stored. Mark lines to memory are calculated for first and last word to be stored. ZZZZ is not used.
00111ZZZZ	J.EXC	The J counter is used to address RF for word to be stored. ZZZZ is not used.
01000BBBB	L-R BBBB	Used during format G to select J counter as RFA address during BDP primitive time. When the most significant bit of BBBB is one and when byte unit is required to process another byte past word boundary, it increments J counter and decrements J length counter. BBBB is described in the end of this appendix.
01001BBBB	R-L BBBB	Identical to 01000BBBB except J counter is decremented at a word boundary.

<u>RGB</u>	<u>Mnemonic</u>	<u>Description</u>
1XXXXXXX	NAME	Identical to their RGA counterparts. Read RFB when reading RFA.
00000YYYY	YYYY	
00001YYYY	YYYY	
0XY00YYYY	YYYY	These codes operate on RGB as they would if
0XY01XXXX	YYYY	X and Y bits were both zero.
0XY10YYYY	YYYY	
0XY10YYYY	YYYY	
0XY11YYYY	YYYY	SBD mux is controlled by XY. See description of SBD control.
01000BBBB	L-R BBBB	Identical in operation to RGA counterparts.
01001BBBB	R-L BBBB	Read RFB, K count and K3, rather than RGA, J count, and J3 in description of RGA.

<u>RGC</u>	<u>Mnemonic</u>	<u>Description</u>
1XXXXXXX	NAME	XXXXXXX is selected by RFA and RFB at write time.
00000YYYY	YYYY	YYYY selects the contents of specified instruction field as address to be selected by RFA and RFB at write time of a particular micrand.
00001YYYY		
00010YYYY		Not used.
00011YYYY		
001000ZZZZ	JK.48	Used during a load multiple streaming type operation to cause J counter to be selected by RFA and RFB at write time of A registers. K counter is selected by RFA and RFB at write time of X registers. Uppermost 16 bits of A registers remain unchanged. ZZZZ is not used.
001100ZZZZ	J.64	Used during a load multiple streaming operation to cause J counter to be selected by RFA and RFB at write time of RF.
001110ZZZZ	K.64	Used during a load multiple streaming operation to cause K counter to be selected by RFA and RFB at write time of RF.
01000CCCC	L-R CCCC	Used during format G to select C counter into WDA for use as an address by RFA and RFB to write into RF. CCCC is described in end of this appendix. During this code, when BDP writes a byte into RF and C3 equals 7 then C address counter increments and C length counter decrements.
01001CCCC	R-L	Identical in operation to 01000CCCC code except C counter decrements when C3 equals 0 and BDP advances write address.

<u>RGC</u>	<u>Mnemonic</u>	<u>Description</u>
00101ZZZZ	JK.64	Identical in operation to 00100ZZZZ code except 16 most significant bits of A registers are written.
01010ZZZZ	SFDR	Strobes stack frame descriptor (SFDR) register with contents of WD mux bits 48 through 63.
01011ZZZZ	SF00	Strobes SFDR with 00FF.
01110mmmm	MAC mmmmm	Causes MAC execution request signal and MAC select code (mmmm) to be valid to MAC until MAC has accepted request. mmmmm is explained in following table.

mmmm

<u>Mnemonic</u>	<u>mmmm</u>	<u>Description</u>
RDC	0000	Load DCI into AD
RO1	0001	Load option installed into AD
RCE	0010	Load CEL into AD
RPT	0011	Load PTM into AD
RPF	0100	Load PFS into AD
RPI	0101	Load PID into AD
REI	0110	Load EID into AD
RCH	0111	Load channel data into AD
	1000	
	1001	
WCE	1010	Load AD into CEL
WPT	1011	Load AD into PTM
WPF	1100	Load AD into PFS
	1101	
WKE	1110	Write keypoint
WCH	1111	Send AD to channel

YYYY

When RGA, RGB, or RGC have a code of 0000YYYY or 0001YYYY, the YYYY section refers to a field in the instruction which is used as a register designator to read or write RF. J, K, and I refer to instruction fields in the instruction for both 60- and 64-bit modes. Not all selections are available in both modes. X offsets the register designator into the X registers of RF.

<u>Mnemonic</u>	<u>YYYY</u>	<u>60-Bit Mode</u>	<u>64-Bit Mode</u>
Aj	0000	00000JJJ	0000JJJJ
Ak	0001	00000KKK	0000KKKK
Ai	0010	00000III	0000IIII
A8j	0011	00001JJJ	XXXXXXX
A8k	0100	00001KKK	XXXXXXX
A8i	0101	00001III	XXXXXXX

<u>Mnemonic</u>	<u>YYYY</u>	<u>60-Bit Mode</u>	<u>64-Bit Mode</u>
Ak/j+ *	0110	XXXXXXXX	0000AAAA
X	0111	XXXXXXXX	XXXXXXXX
Xj	1000	00010JJJ	0001JJJJ
Xk	1001	00010KKK	0001KKKK
Xi	1010	00010III	0001IIII
Bi	1010	00010III	0001IIII
Bj	1000	00010JJJ	0001JJJJ
Bk	1001	00010KKK	0001KKKK
X8j	1011	00011JJJ	0001JJJJ
X8k	1100	00011KKK	0001KKKK
X8i	1101	00011III	0001IIII
Xk/j+ *	1110	XXXXXXXX	0001AAAA
	1111	XXXXXXXX	XXXXXXXX

* In RGA/C field, AAAA refers to K designator plus one. In RGB field, AAAA refers to J designator plus one. In both cases a designator of F added to one will give a result of zero.

When RGA or RGB is OXX10YYYY, the YYYY portion forms the select for the immediate data by either the AD or BD mux.

<u>Mnemonic</u>	<u>YYYY</u>	<u>Immediate Select</u>
ZER	0000	Zero
J	0001	64-bit mode j field instruction designator
JK	0010	64-bit mode jk field instruction designator
JKQ	0011	64-bit mode jkq field sign extended
QSG	0100	64-bit mode q field sign extended
D	0101	64-bit mode d field
CPR	0110	Current P ring # left three
KCY	0111	60-bit mode K field
CJK	1000	60-bit mode j field
E64	1001	Emit 64 register
E13	1010	BRA field from micrand
P	1011	P register
MAP	1100	MAP data
T	1101	Descriptor T field
INS	1110	Instruction length
	1111	

NOTE

All fields are right-justified unless otherwise specified.

BBBB

During an RGA code of 01000BBBB or 01001BBBB, the BBBB is used by the BDP. The most significant bit allows advancement of the streams and the three least significant bits control the validity checking of the byte presently addressed by J3.

<u>Mnemonic</u>	<u>BBB</u>	<u>Validity Check Description</u>
DDS	000	Declare valid
SDS	001	Check for valid type 6 or 8 sign
LDI	010	Check for valid type 2 or 3 or 12 or 13 sign
RDI	100	Check for valid right digit
L.R	101	Check for valid left and right digit
Z.D	110	Check for valid zoned digit
CHC	111	Check for valid combined Hollerith digit

CCCC

During an RGC code of 01000CCCC or 01001CCCC, the CCCC is used by the BDP. The most significant bit of CCCC enables the advance of the write stream. The three least significant bits of CCCC control the result sign in the BDP operations.

<u>Mnemonic</u>	<u>CCC</u>	<u>Validity Check Description</u>
	000	Hold previous result
JSI	001	Use J sign as result sign
NKS	010	Use inverted K sign
XOR	011	Use exclusive OR of K sign and J sign
NIS	100	Use inverted J sign
KSI	101	Use K sign
+VE	110	Set sign positive
-VE	111	Set sign negative

AD MULTIPLEXER SELECTS (ADS)

ADS select equals 111 or 010 and the 60-bit mode bit effects the sign extension in ASX.

<u>Mnemonic</u>	<u>ADS</u>	<u>Description</u>
SIT	000	Choose SIT unconditionally
PSR	001	Process registers are selected into upper 16 bits of AD. This select enables circuit which chooses hard registers rather than RFA according to process register hit circuit. RFA lower 48 is selected unconditionally.

<u>Mnemonic</u>	<u>ADS</u>	<u>Description</u>
RF6	010	Select RF; if read address is same as previous write address, WD mux data is selected by AD mux, rather than register file data. This is referred to as turnaround data.
IMM	011	AD mux selects Immediate Data
TUR	100	AD mux selects WD mux data.
R2B	101	AD mux selects B adder right two.
L1B	110	AD mux selects B adder left one.
RF48	111	AD mux select RFA as in a code of 010.

BD MULTIPLEXER SELECTS (BDS)

When BDS is 01 and 60-bit mode bit is on, sign extension takes place on the input of the shifter i.e., bit 4 is sign extended into bits 0 through 3, and into SBD or bits 0 through 3 in the shifter input.

When BDS is 10, the shifter input sign extends the top 16 bits with bit 4 in 60-bit mode and zero in 64-bit mode.

<u>Mnemonic</u>	<u>BDS</u>	<u>Description</u>
RF6	00	RFB is selected into BD mux. Turnaround is enabled.
RFX	01	RFB is selected into BD Mux. Turnaround is enabled.
RF4	10	RFB data is selected into BD mux.
TUR	11	WD mux data is selected into the BD Mux (equals turnaround).

REGISTER PARTIAL WRITES (RFPW)

During format G, the partial write information is given by C3, the C stream byte write address counter.

<u>Mnemonic</u>	<u>RFPW</u>	<u>Description</u>
NOP	000	Inhibit writing to RF.
R32	001	Write rightmost 32 bits only.
C64	010	Write 64 bits into RF only if execution condition is met.
R48	011	Write into right 48 bits of RF only.
L16	100	Write only left 16 bits into RF. If RGC field had specified JK48 during writes into A registers, write into these bits is inhibited when most significant bit of RFPW field is a one.
XUSED	101	Not used.
L32	110	Write into left 32 bits of RFPW only.
64	111	Unconditionally write 64 bits into RF.

SHIFTER CONTROL (SHC)

<u>Mnemonic</u>	<u>SHC</u>	<u>Description</u>
LE0	000	60-bit mode has no effect; when shift count is greater than 96, data is shifted off. BDS = 01 causes shifter input bits 0 through 3 to be equal to bit 4. BDS = 10 causes output of shifter to be all 1's. Sign injection is into bit 63 and is equal to zero. Shifter shifts data left end off.
LE1	001	60-bit mode has no effect. When shift count is greater than 96, data is shifted off. BDS = 01 and BDS = 10 have same effect as in SHC = 000. Sign injection is into bit 63 and is equal to one. Shifter shifts left, end off.
LES	010	60-bit mode has no effect. When shift count is greater than 96, data is shifted off. BDS = 01 causes top 4 bits of shifter input to be bit 4. BDS = 10 causes output of shifter to be all 1's. Sign injection is into bit 63 and is equal to shifter input bit 0. Shifter shifts left end off.
LCY	011	60-bit mode causes a left cyclic shift 0 mod 60 of shift count. When shift count is greater than or equal to 96, shifter output is all 1's. When BDS = 01, shifter input bits 0 through 3 are set equal to one during 60-bit mode and equal to BD bit 4 during 64-bit mode. When BDS = 10, shifter output is set to all 1's.
RE0	100	60-bit mode has no effect. When shift count is greater than 96, shifted data is shifted off leaving all zeros at output. When BDS = 01, shifter input bits 0 through three is set to BD bit 4. When BDS = 10, shifter input bits 0 through 15 are set to zeros. Shifter otherwise performs a normal, right end off shift with zero injected into upper bits.
RE1	101	60-bit mode has no effect. When shift count is greater than 96, data is shifted off and shifter output is set to all ones. BDS = 01 causes BD bit four to be set as shifter input bits 0 through 3. BDS = 10 causes indeterminate results out of shifter. Otherwise, the shifter performs a normal right end off shift with ones injected into the upper bits.

<u>Mnemonic</u>	<u>SHC</u>	<u>Description</u>
RES	110	When BDS field = 10, 60-bit mode causes BD bit 4 to be used as input to shifter bits 0 through 15. If 60-bit mode is not true, this sign extension is zero. Sign injection, into upper bits, as per bit 0 of sign extended portion specified in the 101 instruction, above. When BDS = 01, the shifter input bits 0 through 3 are BD bit 4 and sign injection is as per bit 0 of sign extended shifter input. Shifter shifts right end off with above rules specifying sign extension and injection.
RCY	111	60-bit mode has no effect. BDS = 01 causes BD bits 0 through 3 to be equal to bit 4 as shifter input. BDS = 10 causes shifter output to be all 1's. Shifter otherwise operates in a normal right cyclic mode.

SHIFT COUNT SELECT (SCS)

The shift amount control selects a seven-bit shift count according to the following table:

<u>Mnemonic</u>	<u>Code</u>	<u>Shift Count Source</u>
ZER	0	Shift count = 0 (Shifter bypassed)
L6	1	Latch bits 12-17
L5	2	Latch bits 13-17
LBP	3	Latch bits 6-11
L7	4	Latch bits 11-17, consider bits 3-10
30	5	Shift count = 48 (10)
RNO	6	Shift count = 0 if RIGHT. Norm false Shift count = 1 if RIGHT Norm true
BRA	7	Emitted BRA field

UNPACKER CONTROL (FP)

Micrand bits 46 and 47 control the A unpacker and B unpacker networks respectively. The micrand bits are valid for the unpacks only during format A; at other times, AD/BD right 18 bits are selected to the S adder.

<u>Mnemonic</u>	<u>Code</u>	<u>Function</u>
STA/STB	0	AD/BD right 18 bits straight to S adder.
A/B	1	AD/BD left 16 (exponent) unpacked per CYBER 170 mode to S adder.

If during formats A or E, the S adder function is 2, 3, or 6 the output of A unpacker or B unpacker is forced to all 1's. See S adder control for explanation. The unpack network is functional at primitive time.

S ADDER LATCH CONTROL (LATCH)

The S adder latch (SAD 2.0) selects either 18 bits of unpacked S adder output or 6 bits of normalization count (right-justified, zero-extended) for the normalization. The latch is controlled by micrand bits 50 and 51 during formats A or E. During other formats, the contents of latch are unaltered. The latch is strobed at P exit time.

<u>Mnemonic</u>	<u>Code</u>	<u>Function</u>
NOP	0	Do not strobe latch
+VE	1	Strobe latch if S adder M.S.B. = 0
S adder	2	Strobe latch unconditionally
NORM	3	Select norm count

BIG ADDER CONTROL (B ADDER)

Micrand bits 39, 40, and 41 define the operation of B adder* (64-bits, 1's or 2's complement) during formats B, C, or D. During other formats the adder function is forced to A+B+C, except during format H (multiply or divide) where B adder is under special control. The adder function is then data dependent.

			CARRY		
			B adder	Function Source	
<u>60-Bit</u>	<u>64-Bit</u>	<u>Code</u>	<u>(L adder)</u>	<u>60-Bit</u>	<u>64-Bit</u>
PLUS64	PLUS	0	A+B+0	B-EAC	0
PLUS96	PLUSC	1	A+B+C	L-COUT	B-COUT F/F
MINUS	MINUS	2	A-B+1	B-EAC	1
MINUS96	MINUSC	3	A-B+C	L-COUT	B-COUT F/F
	BOOLOR	4	A OR B	N/A	N/A
	BOOLAND	5	A AND B	N/A	N/A
	BOOLXOR	6	A EXOR B	N/A	N/A
	INHBIT	7	A AND NOT B	N/A	N/A

During format H, the possible B adder functions are:

Divide/Multiply	(1)	A+B+C
Divide/Multiply	(2)	A-B+C
Multiply only	(3)	A+0

* Note that L adder, the 48-bit ones complement adder, always follows the function of B adder.

SMALL ADDER (S ADDER) CONTROL

Micrand bits 39, 40, and 41 are shared by S adder and B adder. S adder fields are defined during formats A or E; during other formats, the S adder function is forced to plus 2. S adder is an 18-bit ones or twos complement adder.

S ADDER FUNCTION (FORMAT A+E)

<u>Mnemonic</u>	<u>Code</u>	<u>ALU Bits</u>	<u>Function</u>
PLUS1	0	1 1 1	A+B+C 1's complement
MINUS1	1	1 1 0	A-B+C 1's complement
XMITA	2	0 0 0	Pass AD (A and B) B is all 1's forced via unpacker
XMITB	3	0 0 0	Pass BD (A and B) A is all 1's forced via unpacker
PLUS2	4	1 1 1	A+B 2's complement
MINUS2	5	1 1 0	A-B+1 2's complement
PLUSR	6	1 1 1	A+B+C if RT. norm or A and B
MINUA	7	1 0 1	-A+B+1 2's complement

The adder is functional during primitive time.

EXECUTION SENSE FIELD (ESC)

S Adder ESC (Format A+E)

<u>Mnemonic</u>	<u>Code</u>	<u>Condition to Set ESC</u>
HLD	0	Select previous sense condition
EQU	1	A input = B input
NEQ	2	A input not equal to B input
GRT	3	A input is greater than B input
GEQ	4	A input is greater than or equal to B input
EAR	5	Sense for early exception; also saves BD bit 0
SAD	6	S adder output bit 56
RIN	7	Ring bits in AD = Ring bits in BD
SEG	8	Seg bits in AD = Seg bits in BD
ADN	9	AD is normalized

B adder ESC (Format B)

<u>Mnemonic</u>	<u>Code</u>	<u>Condition Set ESC</u>
HLD	0	Select previous executions sense condition
EQF	1	Set on AD equal to BD

<u>Mnemonic</u>	<u>Code</u>	<u>Condition Set ESC</u>
NEF	2	Set on AD not equal to BD
GTF	3	Set on AD greater than BD
GEF	4	Set on AD greater than or equal to BD
EQH	5	Set on equality of the right 32 bits of AD and BD
NEH	6	Set when AD right 32 is not equal to BD right 32
GTH	7	Set when AD right 32 is greater than BD right 32
GEH	8	Set when AD right 32 is greater than or equal to BD right 32
RNO	9	Set if the right norm FF is set
OBI	A	Set if B adder bit zero is a one
AD4	B	Set if AD bit 46 is a one;
	C	set if the retry enable FF is set
	D	
	E	
	F	

For the execution sense conditions where there must be a compare of AD and BD, the adder being used must be in AD-BD mode. The numbers are always to be signed binary numbers.

When the execution sense condition is equal to nine on a format B, a flip-flop sets to retain the result sign for floating point operations and the right norm flip-flop sets to show that the B adder result bits 14 and 15 are not equal. It shows that a right norm is necessary prior to packing the floating point answer.

When the execution sense condition is equal to five on a format A or E, a flip-flop sets when BD bit 0 (or 4 in 60-bit mode) is a one. The packer network is functional at primitive time.

<u>ESC(A)</u>	<u>BD</u>	<u>Result Coefficient Sign for Exponent</u>
0	0	0
0		1
1	0	1
1	1	0

For 60-bit mode, ones complement CYBER 170 exponent and set coefficient sign equal one if ESC(A) equals one. The bias bit is a copy of S adder bit 7 if ESC(A) equals one (result sign is negative).

NORMALIZATION (NORM)

The normalize is defined during format E only by bits 46 and 47 of the micrand. Its functions are:

<u>Mnemonic</u>	<u>Code</u>	<u>Function</u>
64-0	0	64 to bit 0, treat source as signed integer
48-16	1	48 to bit 16
32-32	2	32 to bit 32
64-1	3	64 to bit 0, treat source as unsigned integer

ER MULTIPLEXER (ER MUX)

The ER mux (BAD 2.0) is 64-bits wide and selects one or three sources to the ER register.

<u>Code</u>	<u>Source</u>
00	BD straight
11	ER straight to ER
10	Left 1 (divide functions)
01	Right 2 (multiply functions)

Micrand bits 53 and 54 control the ER mux during formats B and H. During other formats, the ER mux is forced to code equal to 0 (BD straight). The ER mux is active during primitive time.

J3, K3 AND C3 CONTROL

During format F micrand bits 40, 44, and 48 allow the strobe of J3, K3, and C3 respectively.

MUX SELECT (JCNT)

During format F, micrand bits 41, 42, and 43 select the origin of the data to be strobed into the J counter (RF 2.1).

<u>Mnemonic</u>	<u>JCNT</u>	<u>Description</u>
NO.P	000	No-op.
WDX	001	Strobe J count with 0000, and WD mux bits 48 through 51.
WD5	010	Strobe J count with WD mux bits 53 through 60.
-	011	Strobe J count with the C count.
CCNT	100	Strobe J count with the K count.
	101	Not used.
FFH	110	Strobe J count with all ones.

K SELECT (KCNT)

During format F, micrand WD mux bits 46 and 47 select the origin of the data to be strobed into the K counter, shown on RF 2.1.

<u>Mnemonic</u>	<u>KCNT</u>	<u>Description</u>
-	00	Do not strobe K counter.
WDX	01	Strobe K counter with 0001 and WD mux bits 52 through 55.
WDS	10	Strobe K counter with WD mux bits 53 through 60.
CCN	11	Strobe K counter with the C counter.

C MUX SELECT (CCNT)

During format F, micrand bit 50 selects the origin of C mux which is used to initialize the counter (RF 2.1).

<u>Mnemonic</u>	<u>CCNT</u>	<u>Description</u>
-	0	Do not strobe C counter.
WDS	1	Strobe C counter with WD mux bits 53 through 60.

YKW MUX SELECT (YKW)

The YKW field, active during format F, is micrand bits 58 through 60. This field selects the length that may be read by WD mux or strobed into one of the YKW counters (STR 2.1).

<u>Mnemonic</u>	<u>YKW Select</u>	<u>Description</u>
NOP	000	Do not strobe YKW counters; YKW mux selects all 1's.
LDB	001	Select and strobe YKW counters with length determination box output.
LSMO	010	Select and strobe the YKW counter with load/store multiple word length.
LSM1	011	Select and strobe the YKW counters with load/store multiple word length plus one.
Lmux	100	Select L mux into YKW mux and strobe counters.
LENC	101 --	Select C-, K-, or J- length counters into YKW mux and strobe YKW counters with selected length.
LENK	110 →	
LENJ	111 --	

L MUX SELECT (L MUX)

<u>Mnemonic</u>	<u>LMUX</u>	<u>Description</u>
ADL	0	Select the S adder latch
QCO	1	Select the Q counter

COUNTER CO (LENJ, LENK, LENG)

These fields, active during format F, are in bits 53, 54, and 55. They control the strobing of the three length counters.

<u>Mnemonic</u>	<u>LENJ/K/C</u>	<u>Description</u>
NOP	0	No-op
LMUX	1	Strobe LENJ/KC with L mux

MULTIPLY SUBFUNCTION BITS

The subfunction bits work in conjunction with the micrand being treated as an unsigned (that is, positive) coefficient during double precision. The MPY/DIV subfunction bits are active during format H only. The field corresponds to micrand bits 46 and 47.

MUX SELECT AND FUNCTION CODE (XBD)

The XBD field is located in bits 48 to 51 of a format G micrand. The XBD field selects the data path and the function of the byte unit (BDP).

<u>Mnemonic</u>	<u>XBD</u>	<u>Description</u>
RJL	0000	XBD mux selects RJL and XBD mux selects data mux if E mode FF is set.
	0001	Not used.
COM	0010	If XBDP field is equal to 001, function is decimal compare. If not equal to 001, function is byte compare. XBD mux is not used.
RKL	0011	XBD mux selects RKL. If RJL or RKL are latched, this select causes them to unlatch.
	0100	Not used.
DMU	0101	XBD mux selects data mux. Zero field checking is performed on output of data mux.
XAO	0110	XBD mux selects XAO. If EMODE FF is set, XBD mux selects E ROM. Zero field checking is performed on output of data mux.
	0111	Not used.
	1000	Not used.
EDI	1001	XBD mux selects data mux. Zero field checking is performed on output of data mux. Four MSBs into XBD are forced to 0's.
CHC	1010	XBD mux selects combined Hollerith character ROM.

<u>Mnemonic</u>	<u>XBD</u>	<u>Description</u>
SEP	1011	XBD mux selects combined Hollerith character ROM which also contains unpacked separate sign.
ADD	1100	XBD mux selects decimal adder. This is add/subtract function.
MAC	1101	XBD mux selects MAC data; byte unit is functioned for MAC transfers.
SCAN	1110	Bits 5, 6, and 7 of RJL select one bit from RKL. XBD mux is not used.
B-D	1111	XBD mux selects the decimal adder. This selects the byte unit for binary to decimal conversion. During first iteration, RJC selects the binary-decimal circuit; during next iterations, RJC selects D1. RKC selects the X16 circuit.

DATA MUX AND OVERFLOW CONTROL (DMX)

The DMX field is in bits 32 and 33 of a format G micrand. The data mux control and overflow control occupy the same bit locations. The DMX field controls the data mux (BDP 2.1) when the XBD field = 0000 (only when the E MODE FF is set), 0101, 1001, or 1111. The DMX field is used to control overflow checking when the XBD field = 0110, 1010, 1011, or 1100.

DATA MUX CONTROL

<u>Mnemonic</u>	<u>DMX</u>	<u>Description</u>
LDI	00	Data mux selects left digit (4 MSBs) of RJL. If E MODE FF is set, data mux selects translated left digit from data ROM.
RDI	01	Data mux selects right digit (4 LSBs) of RJL. If E MODE FF is set, data mux selects translated right digit from data ROM.
CHC	10	Data mux selects combined Hollerith digit from data ROM.
	11	Not used.

OVERFLOW CONTROL

<u>Mnemonic</u>	<u>DMX</u>	<u>Description</u>
	00	No overflow checking.
ZBI	01	Check for binary nonzero in XAO after LC = 0.
SBI	10	Check for binary nonsign in XAO after LC = 0.
DEC	11	Check for decimal nonzero in XAO after LC = 0.

E MODE FF CONTROL AND SLACK DIGIT (EB)

The EB field is located in bits 34 and 35 of a format G micrand.

<u>Mnemonic</u>	<u>EB</u>	<u>Description</u>
HLD	00	State of E mode FF is not changed.
SET	01	Sets E mode FF.
CLR	10	Clears E mode FF.
DIG	11	Indicates left digit being processed through XAO must be 0. Counters J3 and LJ are inhibited from counting.

SOURCE SIGN CONTROL (GSN)

The GSN field is in bit 37 of a format G micrand. The sign field selects the sign from source data type. When E mode FF is set, the source field sign is taken from the translated double digit sign separate bit in the data ROM.

<u>Mnemonic</u>	<u>GSN</u>	<u>Description</u>
CHC	00	Source field sign is taken from combined Hollerith character sign bit in data ROM.
SEP	01	Source field sign is taken from separate sign bit in data ROM.

J SIGN RECORD (JR)

The JR field is in bits 38 and 39 of a format G micrand.

<u>Mnemonic</u>	<u>JR</u>	<u>Description</u>
	00	State of J sign FF not changed.
JSI	01	J sign FF is strobed with sign per GSN field.
J+V	10	J sign FF set to positive state.
J-V	11	J sign FF set to negative state.

K SIGN RECORD (KR)

The KR field is in bits 40 and 41 of a format G micrand.

<u>Mnemonic</u>	<u>KR</u>	<u>Description</u>
	00	State of K sign FF not changed.
KSI	01	K sign FF is strobed with sign per GSN field.
K+V	10	K sign FF is set to positive state.
K-V	11	K sign FF is set to negative state.

SOURCE FIELD FILL CONTROL (FILL)

The FILL field is in bits 42 and 43 of a format G micrand. This field determines the fill characters to be injected in RJL or RKL when LJ or LK goes to zero.

<u>Mnemonic</u>	<u>FILL</u>	<u>Description</u>
ZER	00	ASCII 0 (30 ₁₆) is used as fill character.
SPA	01	ASCII space (20 ₁₆) is used as fill character.
ONE	10	All ones (FF ₁₆) is used as fill character.
SGN	11	Binary sign (MSB of last byte) extended across byte is used as fill character.

Q COUNT CONTROL (Q, QC)

The Q field is in bit 45 of a format F micrand and the QC field is in bits 44 and 45 of a format G micrand.

<u>Mnemonic</u>	<u>Format</u>	<u>Q</u>	<u>QC</u>	<u>Description</u>
HLD	F,G	0	00	State of Q counter not changed.
CLR	F,G	1	01	Loads Q counter with all 0's. CLR Q counter also initializes following FFs: byte sense, source data = 0, result = 0, overflow, ZF, 10s complement carry, adder carry, and BDP invalid.
INC	G	-	10	Allows Q counter to increment when a byte is stored and during byte compare and byte scan.
ADD	G	-	11	Increments Q counter unconditionally.

XAO MUX CONTROL (XAO)

The XAO field is in bits 46 and 47 of a format G micrand. The XAO field assembles the decimal digits into bytes. The second pass invert FF and EB field is equal to 11 (slack digit) interacts with the XAO field. The second pass invert FF only effects the XAO select when the PAS2 field is equal to 00. Decodes from the XAO field also enable the left and right overflow checkers on XAO.

<u>Mnemonic</u>	<u>XAO</u>	<u>Description</u>
Z.D	00	Four MSBs of XAO (XAO left) select zone 3 ₁₆ . Four LSBs of XAO (XAO right) select four LSBs of RJL. If second pass invert FF is set, XAO right selects 10s complementer. EB equal to 11 has no effect. Only right overflow checker is enabled.

<u>Mnemonic</u>	<u>XAO</u>	<u>Description</u>
L.D	01	XAO left selects four MSBs of RJL and XAO right selects four LSBs of RJL. If second pass invert FF is set, XAO right selects 10s complement. EB = 11 has no effect. If DMX = 01 or 10, both left and right overflow checkers are enabled. Otherwise only right checker is enabled.
D.S	10	XAO left selects 0000 and XAO right selects sign (positive C ₁₆ , negative D ₁₆) second pass invert FF and EB = 11 have no effect. Overflow checkers are disabled.
D.G	11	XAO left selects four LSBs of RJL and XAO right retains previous digit. If second pass invert is set, XAO left selects the 10s complement. If EB = 11, XAO left selects 0000 (EB = 11 overrides second pass invert). Only the left overflow checker is enabled.

BYTE SENSE CONDITIONS (SENZ)

The SENZ field is in bits 52, 53, and 54 of a format G micrand.

<u>Mnemonic</u>	<u>SENZ</u>	<u>Description</u>
	000	No byte sensing.
ZFL	001	Set byte sense FF if source field = 0.
NEB	010	Set byte sense FF if bytes are not equal.
SCA	011	Set byte sense FF if there is a scan hit.
	100	Not used.
	101	Not used.
	110	Not used.
	111	Not used.

BDP EXIT CONTROL (XBDP)

The XBDP field is in bits 55, 56, and 57 of a format G micrand.

<u>Mnemonic</u>	<u>XBDP</u>	<u>Description</u>
LJ=	000	Exit primitive time when LJ = 0.
LJK	001	Exit primitive time when LJ and LK = 0.
LK.	010	Exit primitive time if bytes are not equal or when LK = 0.
NE.	011	Exit primitive time if bytes are not equal or when LJ and LK = 0.
LC=	100	Exit primitive time when LC = 0.
	101	Not used.
LJC	110	Exit primitive time when LJ and LC = 0.
100	111	Exit primitive time after 100 ns.

INVERT CONTROL (PAS2)

The PAS2 field is in bits 58 and 59 of a format G micrand.

<u>Mnemonic</u>	<u>PAS2</u>	<u>Description</u>
2ND	00	Causes XAO to use 10s complementer rather than four LSBs of RJL if second pass invert FF is set.
CLR	01	Clears second pass invert FF.
ADD	10	Causes RJC to select 9's complementer rather than four LSBs of RJL if J sign and K sign FFs are different.
SUB	11	Causes RJC to select 9's complementer rather than four LSBs of RJL if J sign and K sign FFs are same.

J SIGN FIX (JFX)

The JFX field is located in bit 49 of a format F micrand.

<u>Mnemonic</u>	<u>JFX</u>	<u>Description</u>
NOP	00	State of J sign FF not changed.
JSG	01	Set J sign FF to positive state if source field = 0 FF is set.

K SIGN FIX (KFX)

The KFX field is located in bit 56 of a format F micrand.

<u>Mnemonic</u>	<u>KFX</u>	<u>Description</u>
NOP	00	State of K sign FF not changed.
KSG	01	Set K sign FF to positive state if source field = 0 FF is set.

MULTIPLY/DIVIDE SUBFUNCTION (SUBF)

The subfunction fields are decoded off micrand bits 46 and 47 and are active during primitive time. They are sampled during the first format H micrand (multiply or divide). The double precision algorithms and rounded divide use the subfunction bits.

<u>Code</u>	<u>Function</u>
0 MPY	No action.
1 MPY	Treat multiplicand as unsigned (F3).
2 MPY	Save multiplier sign in sign FF (F13); micrand is unsigned.

<u>Code</u>	<u>Function</u>
3 MPY	Use multiplier saved sign in (F23). Recode on first iteration; micrand is unsigned.
4 DIV	Inject zeros.
5 DIV	Inject 1's.
6 DIV	Inject 5 (rounded 60-bit mode divide).
7 DIV	Not assigned.

FLOATING POINT FUNCTION (FP FUNC)

The floating point function fields are decoded off micrand bits 52 through 55 and used with floating point exception sense condition to detect invalid floating point numbers. The field is active at primitive time during format A only. The function addresses a condition matrix unique to this instruction.

<u>Code</u>	<u>Function</u>
0	Compare
1	Add
2	Subtract
3	Multiply
4	64-bit mode divide
5	Convert
6	Normalize
7	60-bit mode divide
8	Branch equal
9	Branch not equal
A	Branch greater than
B	Branch greater than or equal to
C	Add DP
D	Subtract DP
E	Multiply DP
F	Divide DP

FLOATING POINT TRAP ALLOW (TRP)

The floating point trap allow field is decoded off micrand bit 49 during format A micrands.

<u>Code</u>	<u>Function</u>
0	No FP traps allowed.
1	Allow FP traps for overflow or underflow.

During 64-bit mode, user mask register (UMR) bits 10 and 11 inhibit a trap if they are not set and their respective condition occurs. The floating point trap should not be confused with the traps caused by the setting of UCR bits. The operation of this instruction is invisible to the programmer.

If a floating point trap occurs, a five-bit index is used to point to a microcode routine to supply the required canned result for the condition.

FLOATING POINT EXCEPTIONS

The normal flow of a floating point operation may be interrupted in the following ways:

- Early exception
- Floating Point traps

Early Exception

These may occur when operands are checked at the beginning of an instruction. Each exponent is unpacked and encoded via a ROM according to its range as shown below.

<u>64-Bit Mode Exponent</u>	<u>Range</u>	<u>Encoded Value</u>	<u>60-Bit Mode (Octal)</u>
3000-4FFF	+Normal	000	All others
B000-CFFF	-Normal	001	All others
0000-2FFF	+Zero	110	0000
8000-AFFF	-Zero	111	7777
5000-6FFF	+Infinity	100	3777
D000-EFFF	-Infinity	101	4000
7XXX	+Indefinite	010	1777
FXXX	-Indefinite	011	6000

The encoded values are then checked to see if they represent abnormal exponents. If so, a bit is set which is available to microcode as a branch condition.

The encoded values are also used in conjunction with the floating point function field (format A micrand) to address a matrix in ROM which provides a five-bit index to control store. When the microcode senses an exception, it uses the index to branch to a routine which provides a canned result as a final answer instead of doing a normal operation. The appropriate UCR bits are also set.

Floating Point Microtrap

A floating point microtrap may occur at the end of an FP operation when the exponent of the result (S adder output) is checked to see if it is out of range. Checking is enabled by micrand bit 49 in format A and the cleared UMR bits for underflow or overflow. If it is out of range, a microtrap occurs. The FP trap index ROM provides an index to CS which is used by the trap routine (as in early exceptions) to get a canned result. The index is determined from result sign, mode, overflow or underflow, and single or double precision. The UCR bits for underflow or overflow set whether or not the mask bits set.

Late Trap

Late trap is a flip-flop that sets when a trap occurs in the last micrand of an instruction. This flip-flop clears on the next instruction exit if no further trap is occurring. When late trap is set, the instruction designators of the trapped instruction are automatically selected from P buff register in the instruction pipe.

RING-RING SELECTION

During format D, the ring field, micrand bits 49 through 51 control the ring selection into ARVI mux. The selection logic is on the 8TD0 pak, logic chassis location C11. The MAP ring is the larger ring number of the last request ring and its associated R1 value.

<u>Mnemonic</u>	<u>Code</u>	<u>Selection</u>
ADR	0	Select AD ring bits
PRN	1	Select P ring bits
CSU	2	Select CSU ring bits
AD.	3	Select larger of AD and P ring
MAP	4	Select larger of MAP and AD ring
MC.CSU	5	Select larger of MAP and CSU ring
SAVE	6	Save AD ring and select AD ring

EXIT

During format B and D, the exit field defined by bit positions 61 through 63 of the micrand specifies what exit conditions cause the present primitive to exit. During format G the BDP exit condition enables exit; during all the other formats, the primitive exits after 100 ns. The exit field during formats B and D has the following function:

<u>Code</u>	<u>Function</u>
0	Exit after 100 ns.
1	Exit after 150 ns.
2	Exit after 200 ns.
3	MAP EMPTY causes exit.
4	On format D, CSU ready causes exit; on format B, exit after 100 ns.
5	MAP is quiet causes exit.
6	Exit on last word of a load multiple.
7	To exit during a store multiple after last word is stored, on a format G when BDP is ready to exit, or on a MAC transfer when MAC sends an accept; code is forced by hardware and need not be set by microprogrammer.

APPENDIX F

INSTRUCTION EXECUTION TIMES

=====

This section specifies the instruction execution times of CYBER 180 Models 810 and 830 CPU when in 64- and 60-bit modes shown in tables F-1 and F-2. Times are given in nanoseconds. The assumptions relating to performance are stated in this manual.

No allowance has been made for central memory conflicts or for register addressing conflicts between adjacent instructions. No overlap of these instruction timings occurs. Program timings may be developed by simple addition of instruction execution times.

64-BIT MODE INSTRUCTION TIMINGS

Terminology:

- E - Number of page table entries searched times 2100 nS
- B - Number of bytes processed or compared before exit times 100 nS
- MA - Memory access time. Model 810: 1350 nS, Model 830: 550 nS
- JD - Number of j operand digits (or 8-bit bytes) times 100 nS
- KD - Number of k operand digits (or 8-bit bytes) times 100 nS
- M - Number of edit micro-ops times 100 nS
- Mwd - Number of 64-bit words occupied by edit mask times 100 nS
- Jwd - Number of 64-bit words occupied by J field times 100 nS
- Kwd - Number of 64-bit words occupied by K field times 100 nS
- [] - Use max value included, that is, [KD.JD] larger of KD or JD
- SC - Scale shift count times 100 nS
- T - Branch taken
- NT - Branch not taken
- MS - Number of byte collations required times 100 nS (reference 85).
- N - Number of registers to be transferred
- Nx - Number of X registers to be transferred
- HR - Time penalty for a map miss is $(4.0 + 1.7E)$ microseconds where E is number of entries searched.
- x - Number of 64 word blocks.
- Dep - (Data) Dependent - on operands, normalization, etc.

Notes appear in the tables.

NOTE

These execution times are approximations only and subject to change without notice. Accurate timings can come only from benchmark tests. Control Data Corporation is not responsible for assumptions made based on the times listed here.

TABLE F-1. MODEL 810 CP INSTRUCTION TIMING (Sheet 1 OF 3)

Instruction Code	Description	Execution Time (Nanoseconds)	Timing Notes
010	Return Jump	1100	
011	Read ECS	$3600 + x(MA + 27500)$	
012	Write ECS	$3600 + x(MA + 27500)$	
013	Exchange Jump	$14000 + MA$	
014	Read UEM	$600 + MA$	
015	Write to UEM	600	
016	Read Free Running Counter	$300 + MA$	
02	Branch to Bit K	1300	
030	Branch $X_j = 0$	450, 1500	1
031	Branch $X_j \neq 0$	450, 1500	1
032	Branch $X_j > 0$	350, 1500	1
033	Branch $X_j < 0$	350, 1500	1
034	Branch X_j in range	450, 1500	1
035	Branch X_j out range	450, 1500	1
036	Branch X_j definite	450, 1500	1
037	Branch X_j indefinite	450, 1500	1
04	Branch $B_i = B_j$	450, 1500	1
05	Branch $B_i \neq B_j$	450, 1500	1
06	Branch $B_i \geq B_j$	450, 1500	1
07	Branch $B_i < B_j$	450, 1500	1
10	Copy X_j to X_i	200	
11	Logical Product	100	
12	Logical Sum	100	
13	Logical Difference	100	
14	Complement	100	
15	Logical Product, Comp	100	
16	Logical Sum, Comp	200	
17	Logical Diff, Comp	200	
20	Shift Left Circular	200	
21	Shift Right	200	
22	Shift X_k by B_j	400, 500	2
23	Shift X_k by B_i Comp	400, 500	2
24	Normalize	650	
25	Round & Normalize	850	
26	Unpack	350	
27	Pack	250	
30	Floating Sum	2250	

Timing Notes:

1. First time shown if branch not taken; second time shown if branch taken.
2. First time shown if left shift; second time shown if right shift.
Type of shift depends on the sign.

TABLE F-1. MODEL 810 CP INSTRUCTION TIMING (Sheet 2 OF 3)

Instruction Code	Description	Execution Time (Nanoseconds)	Timing Notes
31	Floating Diff	2250	
32	Floating DP Sum	2400	
33	Floating DP Diff	2400	
34	Floating Sum Rounded	2300	
35	Floating Diff Rounded	2300	
36	Integer Sum	350	
37	Integer Diff	350	
40	Floating Product	6800 (Data Dep \pm 400)	
41	Floating Product Rounded	7150 (Data Dep \pm 400)	
42	Floating DP Product	6350 (Data Dep \pm 400)	
43	Form Mask	200	
44	Floating Divide	8600	
45	Floating Divide Rounded	8600	
46	No Operation	100	
464	Move Indirect	LW > 1: 700 + MA(2MA + 700) + [(LW-1)*(MA + 300) + MA]	3
		LW = 1: 700 + MA(2MA + 700) + MA	3
465	Move Direct	800 + MA(2MA + 700) + [(LW-1)*(MA + 300) + MA]	3
466	Compare Collated	2700 + 2MA + LW*3MA + MISS	3, 4
467	Compare Uncollated	2700 + 2MA + LW*2MA + 70047	3, 4
47	Population Count	900 + (pop*400)	
50	Ai = (Aj) + K	150	i = 0
50	Ai = (Aj) + K	650 + MA	i = 1-5
50	Ai = (Aj) + K	750	i = 6,7
51	Ai = (Bj) + K	150	i = 0
51	Ai = (Bj) + K	650 + MA	i = 1-5
51	Ai = (Bj) + K	750	i = 6,7
52	Ai = (Xj) + K	150	i = 0
52	Ai = (Xj) + K	650 + MA	i = 1-5
52	Ai = (Xj) + K	750	i = 6,7
53	Ai = (Xj) + (Bk)	150	i = 0
53	Ai = (Xj) + (Bk)	650 + MA	i = 1-5
53	Ai = (Xj) + (Bk)	750	i = 6,7
Timing Notes:			
3. LW = Number of destination fields involved.			
4. MISS = 700 for no miscompare MISS = 4000 + MA + 3300 (number of miscompare -1)			

TABLE F-1. MODEL 810 CP INSTRUCTION TIMING (Sheet 3 OF 3)

Instruction Code	Description	Execution Time (Nanoseconds)	Timing Notes
54	$A_i = (A_j) + (B_k)$	150	$i = 0$
54	$A_i = (A_j) + (B_k)$	650 + MA	$i = 1-5$
54	$A_i = (A_j) + (B_k)$	750	$i = 6,7$
55	$A_i = (A_j) - (B_k)$	150	$i = 0$
55	$A_i = (A_j) - (B_k)$	650 + MA	$i = 1-5$
55	$A_i = (A_j) - (B_k)$	750	$i = 6,7$
56	$A_i = (B_j) + (B_k)$	150	$i = 0$
56	$A_i = (B_j) + (B_k)$	650 + MA	$i = 1-5$
56	$A_i = (B_j) + (B_k)$	750	$i = 6,7$
57	$A_i = (B_j) - (B_k)$	150	$i = 0$
57	$A_i = (B_j) - (B_k)$	650 + MA	$i = 1-5$
57	$A_i = (B_j) - (B_k)$	750	$i = 6,7$
60	$B_i = (A_j) + K$	150	
61	$B_i = (B_j) + K$	150	
62	$B_i = (X_j) + K$	150	
63	$B_i = (X_j) + (B_k)$	150	
64	$B_i = (A_j) + (B_k)$	150	
65	$B_i = (A_j) - (B_k)$	150	
66	$B_i = (B_j) + (B_k)$	150	
660	Read CM at (X_k) to X_j	300 + MA	
67	$B_i = B_j - B_k$	150	
670	Write X_j into CM at (X_k)	300	
70	$X_i = (A_j) + K$	150	
71	$X_i = (B_j) + K$	150	
72	$X_i = (X_j) + K$	150	
73	$X_i = (X_j) + (B_k)$	150	
74	$X_i = (A_j) + (B_k)$	150	
75	$X_i = (A_j) - (B_k)$	150	
76	$X_i = (B_j) + (B_k)$	150	
77	$X_i = (B_j) - (B_k)$	150	

TABLE F-2. MODEL 830 CP INSTRUCTION TIMING (Sheet 1 OF 3)

Instruction Code	Description	Execution Time (Nanoseconds)	Timing Notes
010	Return Jump	900	
011	Copy from UEM	3400 + X(MA + 14700)	
012	Copy to UEM	3400 + X(MA + 14700)	
013	Exchange Jump	10800 + MA	
014	Read UEM	600 + MA	
015	Write UEM	600	
016	Read Free Running Counter	300 + MA	
02	Branch to Bi + K	1100	
030	Branch Xj = 0	450, 1300	1
031	Branch Xj \neq 0	450, 1300	1
032	Branch Xj > 0	350, 1300	1
033	Branch Xj < 0	350, 1300	1
034	Branch Xj in range	450, 1300	1
035	Branch Xj out range	450, 1300	1
036	Branch Xj def	450, 1300	1
037	Branch Xj indef	450, 1300	1
04	Branch Bi = Bj	450, 1300	1
05	Branch Bi \neq Bj	300, 1400	1
06	Branch Bi \geq Bj	300, 1400	1
07	Branch Bi < Bj	300, 1400	1
10	Copy Xj to Xi	100	
11	Logical Product	100	
12	Logical Sum	100	
13	Logical Difference	100	
14	Complement	100	
15	Logical Product, Comp	100	
16	Logical Sum, Comp	200	
17	Logical Diff, Comp	200	
20	Shift Left Circular	200	
21	Shift Right	200	
22	Shift Xk by Bj	400, 500	2
23	Shift Xk by Bi Comp	400, 500	2
24	Normalize	650	
25	Round & Normalize	850	
26	Unpack	350	
27	Pack	250	
30	Floating Sum	800	
31	Floating Diff	800	

Timing Notes:

1. First time shown if branch not taken; second time shown if branch taken.
2. First time shown if left shift; second time shown if right shift.
Type of shift depends on the sign.

TABLE F-2. MODEL 830 CP INSTRUCTION TIMING (Sheet 2 OF 3)

Instruction Code	Description	Execution Time (Nanoseconds)	Timing Notes
32	Floating DP Sum	950	
33	Floating DP Diff	950	
34	Floating Sum Rounded	850	
35	Floating Diff Rounded	850	
36	Integer Sum	100	
37	Integer Diff	100	
40	Floating Product	4750 (Data Dep \pm 400)	
41	Floating Product Rounded	5100 (Data Dep \pm 400)	
42	Floating DP Product	4300 (Data Dep \pm 400)	
43	Form Mask	200	
44	Floating Divide	6550 (Data Dep \pm 50)	
45	Floating Divide Rounded	6550 (Data Dep \pm 50)	
46	No Operation	100	
464	Move Indirect	LW > 1: 700 + MA(2MA + 700) + [(LW-1)*(MA + 300) + MA]	3
		LW = 1: 700 + MA(2MA + 700) + MA	3
465	Move Direct	800 + MA(2MA + 700) + [(LW-1)*(MA + 300) + MA]	3
466	Compare Collated	2700 + 2MA + LW*2MA + MISS	3, 4
467	Compare Uncollated	2700 + 2MA + LW*2MA + 70047	3, 4
47	Population Count	900 + (pop*400)	
50	Ai = (Aj) + K	150	i = 0
50	Ai = (Aj) + K	350 + MA	i = 1-5
50	Ai = (Aj) + K	450	i = 6,7
51	Ai = (Bj) + K	150	i = 0
51	Ai = (Bj) + K	350 + MA	i = 1-5
51	Ai = (Bj) + K	450	i = 6,7
52	Ai = (Xj) + K	150	i = 0
52	Ai = (Xj) + K	350 + MA	i = 1-5
52	Ai = (Xj) + K	450	i = 6,7
53	Ai = (Xj) + (Bk)	150	i = 0
53	Ai = (Xj) + (Bk)	350 + MA	i = 1-5
53	Ai = (Xj) + (Bk)	450	i = 6,7

Timing Notes:

3. LW = Number of destination fields involved.

4. MISS = 700 for no miscompare

MISS = 4000 + MA + 3300 (number of miscompare -1)

TABLE F-2. MODEL 830 CP INSTRUCTION TIMING (Sheet 3 OF 3)

Instruction Code	Description	Execution Time (Nanoseconds)	Timing Notes
54	$A_i = (A_j) + (B_k)$	150	$i = 0$
54	$A_i = (A_j) + (B_k)$	350 + MA	$i = 1-5$
54	$A_i = (A_j) + (B_k)$	450	$i = 6,7$
55	$A_i = (A_j) - (B_k)$	150	$i = 0$
55	$A_i = (A_j) - (B_k)$	350 + MA	$i = 1-5$
55	$A_i = (A_j) - (B_k)$	450	$i = 6,7$
56	$A_i = (B_j) + (B_k)$	150	$i = 0$
56	$A_i = (B_j) + (B_k)$	350 + MA	$i = 1-5$
56	$A_i = (B_j) + (B_k)$	450	$i = 6,7$
57	$A_i = (B_j) - (B_k)$	150	$i = 0$
57	$A_i = (B_j) - (B_k)$	350 + MA	$i = 1-5$
57	$A_i = (B_j) - (B_k)$	450	$i = 6,7$
60	$B_i = (A_j) + K$	150	
61	$B_i = (B_j) + K$	150	
62	$B_i = (X_j) + K$	150	
63	$B_i = (X_j) + (B_k)$	150	
64	$B_i = (A_j) + (B_k)$	150	
65	$B_i = (A_j) - (B_k)$	150	
66	$B_i = (B_j) + (B_k)$	150	
660	Read CM at (X_k) to X_j	300 + MA	
67	$B_i = B_j - B_k$	150	
670	Write X_j into CM at (X_k)	300	
70	$X_i = (A_j) + K$	150	
71	$X_i = (B_j) + K$	150	
72	$X_i = (X_j) + K$	150	
73	$X_i = (X_j) + (B_k)$	150	
74	$X_i = (A_j) + (B_k)$	150	
75	$X_i = (A_j) - (B_k)$	150	
76	$X_i = (B_j) + (B_k)$	150	
77	$X_i = (B_j) - (B_k)$	150	

COMMENT SHEET

MANUAL TITLE: CYBER 180 Models 810 and 830 Theory of Operation

PUBLICATION NO.: 60469460

REVISION: B

NAME: _____

COMPANY: _____

STREET ADDRESS: _____

CITY: _____ STATE/PROV.: _____ POSTAL CODE _____

This form is not an order blank.

Control Data Corporation welcomes your evaluation of this manual. Please indicate any errors, suggested additions or deletions, or general comments below (please include page number references).

This comment sheet becomes a self-mailer when the binding edge is cut off and it is folded along the dashed lines on the reverse side.

fold

fold

Place correct
postage here.

CONTROL DATA CANADA
Toronto Publications (210)
1855 Minnesota Court
MISSISSAUGA, Ontario, CANADA
LSN 1K7

fold

fold

CORPORATE HEADQUARTERS, P.O. BOX 0, MINNEAPOLIS, MINN. 55440
SALES OFFICES AND SERVICE CENTERS IN MAJOR CITIES THROUGHOUT THE WORLD

LITHO IN U.S.A



CONTROL DATA CORPORATION